Testing a Class of Methods for Solving Minimization Problems with Simple Bounds on the Variables

A.R. Conn, N.I.M. Gould, and Ph. L. Toint

Research Report CS-86-45

October 1986

This report is simultaneously issued at the University of Waterloo, and the FUNDP at Namur.

Testing a Class of Methods for Solving Minimization Problems with Simple Bounds on the Variables.

by Andrew R. Conn †, Nicholas I. M. Gould ‡ and Philippe L. Toint §

6th October 1986

Abstract. We describe the results of a series of tests upon a class of new methods of trust region type for solving the simple bound constrained minimization problem. The results are encouraging and lead us to believe that the methods will prove useful in solving large problems.

† Department of Computer Science, University of Waterloo, Ontario, Canada. The research of this author was made possible by the NSERC grant A8639.

‡ Computer Science and Systems Division, A.E.R.E. Harwell, Oxford, England.

§ Department of Mathematics, Facultés Universitaires ND de la Paix, B–5000, Namur, Belgium.

KeyWords. Trust regions, optimization with simple bounds, numerical results.

Abreviated title. Testing methods for bounded minimization.

1. Introduction.

We are concerned with solving the simple bound constrained minimization problem

$$\begin{aligned} &\underset{x \in R^n}{\text{minimize}} \quad f(x) \\ &\text{subject to the simple bounds} \quad l \leq x \leq u \end{aligned} \tag{1.1}$$

by a method of the trust region type. In this problem, $f(x)$ is assumed to be sufficiently smooth, l and u are fixed vectors and the inequalities are taken componentwise.

Such problems arise quite naturally in a number of different circumstances. Indeed, some authors (see, Gill, Murray and Wright, 1981) maintain that the variables for the vast majority of optimization problems can only be considered meaningfull within specific intervals and consequently should be solved with simple bound constraints.

Many algorithms for solving (1.1) have been proposed. Early methods were of the active set variety in which a sequence of problems, for which a subset of the variables (the active set) are fixed at bounds and the objective function minimized with respect to the remaining variables, are solved. Such algorithms typically use line searches to force convergence and both direct (Fletcher and Jackson, 1974, Gill and Murray, 1976) and iterative (O'leary, 1980) methods have been suggested for solving the linear systems which arise during each iteration. A significant drawback of such methods for large-scale problems appears to be that the active sets can only change slowly and many iterations are necessary to correct for a bad initial choice. More recently, methods which allow a rapid change of the active set to occur have been proposed (Bertsekas, 1982, Dembo and Tulowitzki, 1983); these methods perform linesearches along a search direction which is 'bent' to keep the iterates within the feasible region. Although a convergence theory for such methods can be given for convex problems, it is not obvious how to adapt the theory for more general problems.

In this paper, we propose a new algorithm for which we have already produced a general convergence theory (Conn, Gould and Toint, 1986) and which allows rapid changes in the active set. At each iteration of the algorithm, we define a (quadratic) approximation to the objective function, and a region surrounding the current iterate in which we believe this approximation to be adequate. The algorithm then finds a feasible candidate for the next iterate in this region. This candidate is chosen so as to give a sufficient reduction in the value of the (quadratic) model of the objective function. If the function value calculated at this point matches its predicted value closely enough, the new point is accepted as the next iterate and the trust region is possibly enlarged; otherwise the point is rejected and the trust region size decreased.

A particularly attractive aspect of the algorithm proposed is that, by generalizing the standard notion of a Cauchy point to accommodate the bound constraints, in what seems to us a natural manner, one is able to extend the very general known convergence results for trust region methods applied to unconstrained problems (see Moré). Moreover, the framework presented is well suited to large dimensional problems and can be used in conjunction with partitioned secant updating techniques on the general class of partially separable problems (Griewank and Toint, 1982a, b). However, the issues involved in solving large scale problems are by no means trivial, and the design of sophisticated software for such problems is a major task that we postpone for a future paper. The purpose of the current work is to demonstrate the viability of the framework proposed

(and whose convergence is studied) in our previous paper. Consequently, the numerical results presented below are for small dense problems – the largest containing 45 variables.

The paper is organised as follows. In §2 we give a general introduction to the framework of our algorithms and we discuss the issues involved in determining our generalization of the Cauchy point in §3. The specific algorithm used in our tests is then given in §4. In §5 details of our testing and the numerical results obtained are presented and a general discussion of some of the directions for future research follows in §6. A detailed list of the test problems that we have used is given in an appendix.

2. The basic algorithm.

As we have already stated, our algorithm is essentially of the model-trust region variety. The description we give here is a special case of the more general framework presented by Conn, Gould and Toint (1986). We need to define the following concepts. The set of all points x for which $l \leq x \leq u$ is the *feasible region* for the problem (1.1) and any point x in the feasible region is *feasible*. Let a and b be any two vectors for which $a \leq b$. The *active set with respect to the vectors a and b at the point x*, $I(x, a, b)$, is the index set $\{i$ for which $x_i \leq a_i$ or $x_i \geq b_i$ (or both)$\}$. In particular, we shall refer to $I(x, l, u)$ as *the active set* at x and denote it by $I(x)$. We now define the projection operator $P[\cdot]$ (componentwise) by

$$P[x, a, b]_i = \begin{cases} a_i & \text{if } x_i \leq a_i \\ b_i & \text{if } x_i \geq b_i \\ x_i & \text{otherwise} \end{cases}$$

For brevity we shall write $P[x] = P[x, l, u]$ and will refer to $P[x]$ as the *projection of x into the feasible region.*

At the k–th stage of the algorithm, we suppose that we have a feasible point $x^{(k)}$, the gradient, $g^{(k)}$, of $f(x)$ at $x^{(k)}$ and a suitable symmetric approximation $B^{(k)}$ to the Hessian matrix of the objective function at this point. Furthermore we require a scalar $\Delta^{(k)}$ which represents a bound upon displacements around $x^{(k)}$ within which we believe that a second order approximation,

$$m^{(k)}(x^{(k)} + s) \equiv f(x^{(k)}) + g^{(k)T} s + \tfrac{1}{2} s^T B^{(k)} s, \tag{2.1}$$

to the objective function will effectively agree with the true function (the trust region). A trial point $\bar{x}^{(k+1)} = x^{(k)} + s^{(k)}$ is constructed by finding an approximation to the solution of the trust region problem

$$\begin{aligned} &\underset{x \in R^n}{\text{minimize}} && m^{(k)}(x) \\ &\text{subject to the simple bounds} && l \leq x \leq u \\ &\text{and the trust region constraint} && \|x - x^{(k)}\| \leq \Delta^{(k)}, \end{aligned} \tag{2.2}$$

where $\|.\|$ is a suitably chosen norm. The acceptance of the trial point as an improvement and the modification of $\Delta^{(k)}$ are treated in essentially the same way as in trust region methods for unconstrained problems (see, Moré, 1983). That is we compute the ratio of the achieved to the predicted reduction of the objective function,

$$\rho^{(k)} = (f(x^{(k)}) - f(x^{(k)} + s^{(k)})) / (f(x^{(k)}) - m^{(k)}(x^{(k)} + s^{(k)})) \tag{2.3}$$

and set

$$x^{(k+1)} = \begin{cases} x^{(k)} + s^{(k)} & \text{if } \rho^{(k)} > \mu \\ x^{(k)} & \text{if } \rho^{(k)} \leq \mu \end{cases}$$

and

$$\Delta^{(k+1)} = \begin{cases} \gamma_0 \Delta^{(k)} & \text{if } \rho^{(k)} \leq \mu \\ \Delta^{(k)} & \text{if } \mu < \rho^{(k)} < \eta \\ \gamma_2 \Delta^{(k)} & \text{if } \rho^{(k)} \geq \eta, \end{cases} \tag{2.4}$$

where $\gamma_0 < 1 < \gamma_2$, μ and η are appropriate numbers. It remains to describe our approximate solution of (2.2).

It is convenient to attempt to solve (2.2) in the case where we choose the infinity norm for the trust region constraint for then the shape of the trust region is aligned with the simple bounds (see, for example, Fletcher, 1980). Thus we replace the constraints in (2.2) by the "box" constraints

$$\max (l_i, x_i^{(k)} - \Delta^{(k)}) \equiv l_i^{(k)} \leq x_i \leq u_i^{(k)} \equiv \min (u_i, x_i^{(k)} + \Delta^{(k)}) \tag{2.5}$$

for $i = 1, \ldots, n$. In order to satisfy our global convergence theory, we need only find a feasible point within the trust region at which the value of the model function is no larger than its value at the Generalized Cauchy Point (GCP) (see Conn, Gould and Toint, 1986). The GCP is defined as the first local minimizer of the univariate function

$$q^{(k)}(t) \equiv m^{(k)}(P[x^{(k)} - t\, g^{(k)}, l^{(k)}, u^{(k)}]). \tag{2.6}$$

That is, the first local minimizer of the model function, along the piecewise linear arc defined by projecting the steepest descent direction into the region (2.5). The calculation of the GCP may be performed extremely efficiently as we shall shortly show. We note that the combinatorial part of our algorithm (the determination of which variables are to be fixed at one of their bounds during the current iteration) takes place in finding the GCP. This computation allows us to add and drop many bounds from the active set during a single iteration if this is desirable. As we have already remarked, this is important for large-scale problems.

In order to provoke a fast asymptotic rate of convergence of the method, we normally require a better approximation to the minimizer of (2.2) than the GCP. Consequently we suppose that the GCP is $x^{(k)C}$ and that the active set with respect to $l^{(k)}$ and $u^{(k)}$ at $x^{(k)C}$ is $I(x^{(k)C}, l^{(k)}, u^{(k)})$. We now apply the conjugate gradient algorithm, starting from $x = x^{(k)C}$, to the problem

$$\underset{x \in R^n}{\text{minimize}} \; m^{(k)}(x), \tag{2.7}$$

with the restriction that *the variables in the set* $I(x^{(k)C}, l^{(k)}, u^{(k)})$ *remain fixed throughout the process.* Under a strict complementary slackness assumption, this strategy is sufficient to ensure that the set of constraints active at the solution (the *correct* active set) is identified after a finite number of iterations (see Conn, Gould and Toint, 1986). The

convergence of the algorithm can thereafter by analysed as that of a purely unconstrained method. The conjugate gradient algorithm is terminated at the point $\bar{x}$ if

(i) the norm of the restricted gradient of the model function, that is the vector whose components are those of the gradient of $m^{(k)}(x)$ at $x = \bar{x}$ not indexed by $I(x^{(k)c}, l^{(k)}, u^{(k)})$, is less than $\eta^{(k)}$ for some $\eta^{(k)}$;

(ii) one or more of the unrestricted variables violates one of the bounds (2.5) ($\bar{x}$ is then the point at which the offending bound(s) is (are) encountered); or

(iii) an excessive number of iterations has been taken.

In all cases, the final point reached is the new trial point, i.e. ($\bar{x}^{(k+1)} = \bar{x}$). The rule (i) has already been considered by Toint(1981) and Steihaug(1983) for unconstrained minimization algorithms and are known to be useful for large-scale computations. Rule (ii) has three purposes. The first is to provide a natural stopping rule in the event that non-positive curvature is encountered for the model. The second is to maintain feasibility of the iterates. The last is to avoid the expense of restarting the conjugate gradient iteration when a bound is encountered.

Once the correct active set has been determined, the asymptotic rate of convergence of the method will be controlled by the accuracy to which we attempt to solve (2.7). A superlinear rate of convergence can be assured provided that (a) the ratio of the norm of the restricted gradient at the final point to that at $x^{(k)}$ tends to zero as the iterates approach a Kuhn–Tucker point for the problem and (b) the matrices $B^{(k)}$ approach the true second derivative matrix at the solution in the direction that $x^{(k)}$ approaches the solution (see, Dembo, Eisenstat and Steihaug, 1982, Dennis and Schnabel, 1983). A suitable choice for $\eta^{(k)}$ in order to satisfy (a) is given by

$$\eta^{(k)} = \min\,(0.1, \sqrt{\|\bar{g}^{(k)}\|})\,\|\bar{g}^{(k)}\|, \tag{2.8}$$

where $\bar{g}^{(k)}$ is the projected gradient, $P[x^{(k)} - g^{(k)}] - x^{(k)}$, at $x^{(k)}$. The property (b) is certainly true if exact second derivatives are computed; the situation for secant updating formulae is closely related to that given by Steihaug (1983). In this case we must be careful that our updates obey the condition

$$\sum_{k=0}^{\infty} 1/(1 + \max_{0\leq i\leq k} \|B^{(i)}\|) = \infty \tag{2.9}$$

required in our global convergence theory. Such a condition is satisfied if, for instance,

$$\|B^{(k)}\| \leq c_1 + k\,c_2. \tag{2.10}$$

for some positive constants c_1 and c_2.

3. The Generalized Cauchy Point.

In order to find the Generalized Cauchy Point, we need to be able to determine the first local minimizer of a quadratic function along a piecewise linear arc. In this section we describe an efficient algorithm for this calculation.

We shall consider a piecewise linear arc of the form

$$x(t) = x^0 + d(t), \text{ where } d(t)_i = \begin{cases} t\,d_i & \text{if } t \leq t_{j_i} \\ t_{j_i}\,d_i & \text{otherwise} \end{cases} \tag{3.1}$$

where the breakpoints $0 = t_0 < t_1 < \ldots < t_m$ and the indices $0 \leq j_i \leq m$ are well defined and may be calculated as required. We shall adopt the convention that $j_i = 0$ if $d_i = 0$ and note that $d(t)$ is independent of t for all $t > t_m$. Notice that the arc

$$P[x^{(k)} - t\, g^{(k)}, l^{(k)}, u^{(k)}]$$

is exactly of this form, a breakpoint occuring whenever a variable reaches one of its bounds.

Our method for finding the GCP is simply to consider the intervals $[t_j, t_{j+1}]$ in order of increasing j until the one containing the GCP is located. We need only calculate t_{j+1} when the first j intervals have been examined and rejected. In order to calculate the GCP, we shall be concerned with the behaviour of the quadratic function

$$m(x) = f + (x - x^0)^T g + \tfrac{1}{2}(x - x^0)^T B\,(x - x^0) \tag{3.2}$$

for points lying on (3.1). In particular we shall be concerned with its behaviour between adjacent pairs of breakpoints. If we define $x^j = x(t_j)$, we can express (3.1) in the interval $[t_j, t_{j+1}]$ as

$$x(t) = x^j + \Delta t\, d^j, \text{ where } d_i^j = \begin{cases} d_i & \text{if } j > j_i \\ 0 & \text{otherwise} \end{cases} \tag{3.3}$$

and $\Delta t = t - t_j$. Hence, defining

$$f_j = f + g^T z^j + \tfrac{1}{2} z^{jT} B z^j,$$

$$f_j' = g^T d^j + d^{jT} B z^j$$

$$\text{and } f_j'' = d^{jT} B d^j,$$

where $z^j = x^j - x^0$, and combining (3.2) and (3.3), we obtain

$$m(x(t)) = f_j + \Delta t\, f_j' + \tfrac{1}{2}\Delta t^2 f_j'',$$

for all $t = t_j + \Delta t$ in the interval $[t_j, t_{j+1}]$. It is then straightforward to deduce that the GCP lies at $t_j - f_j'/f_j''$ provided that this point lies in the interval (t_j, t_{j+1}) and that $f_j'' > 0$, lies at t_j if $f_j' \geq 0$ and lies at or beyond t_{j+1} in all other cases. We need now only be concerned with calculating f_j' and f_j''.

Let $J_j = \{i \text{ such that } j_i = j\}$. Then

$$d^j = d^{j-1} - \sum_{i \in J_j} d_i\, e^i,$$

where e^i is the i–th column of the identity matrix. We may then obtain the identities

$$\begin{aligned} f_j' &= f_{j-1}' + \Delta t_{j-1} f_{j-1}'' - b^{jT} x^j - \sum_{i \in J_j} d_i\, g_i^0 \\ \text{and} \quad f_j'' &= f_{j-1}'' + b^{jT}\Big(\sum_{i \in J_j} d_i\, e^i - 2b^j\Big), \end{aligned} \tag{3.4}$$

where $g^0 = g - Bx^0$, $\Delta t_{j-1} = t_j - t_{j-1}$ and

$$b^j = B\Big(\sum_{i \in J_j} d_i\, e^i\Big) = \sum_{i \in J_j} d_i\, (B\, e^i). \tag{3.5}$$

Notice that the recurrences (3.4) require a single matrix-vector product (3.5) and two vector inner products. Moreover, the matrix-vector product only involves columns of B

indexed by J_j; as it is usual for $|J_j|$ to be small (typically 1) the product can normally be formed very efficiently. In the worst case, when we have to run through all of the breakpoints to find the GCP, each column of B will be accessed once. The recurrences are initialized with the values

$$f_0' = g^T d^0 \quad \text{and} \quad f_0'' = d^{0\,T} B d^0.$$

4. The specific algorithm.

We are now in a position to specify our algorithm in almost its entirety. The only part that we do not give here are the details of how the second derivative approximations $B^{(k)}$ are formed and updated. Details of these calculations are given in §5.

STEP 0 [Initialization]

The feasible starting point $x^{(0)}$, the function value $f(x^{(0)})$ and the gradient value $g^{(0)}$ are given, as well as an initial trust region radius, $\Delta^{(0)}$, and $B^{(0)}$, an initial symmetric approximation to the Hessian matrix of $f(x)$ at the starting point. The positive constants $\gamma_0 < 1 < \gamma_2$, μ, η, and ε are specified. Set $k = 0$.

STEP 1 [Test for convergence]

Compute the projected gradient $\tilde{g}^{(k)} = P[x^{(k)} - g^{(k)}] - x^{(k)}$. If $\|\tilde{g}^{(k)}\| < \varepsilon$, STOP.

STEP 2 [Find the GCP]

Calculate the bounds $l^{(k)}$ and $u^{(k)}$ from equation (2.5). Find the Generalized Cauchy Point, $x^{(k)c}$ (using the algorithm outlined in §3) as follows:

STEP 2.0 [Initialization]

Set

$$x = x^{(k)}, \quad g = g^{(k)} - B^{(k)} x^{(k)},$$
$$d = P[x^{(k)} - g^{(k)}, l^{(k)}, u^{(k)}] - x^{(k)},$$
$$f' = g^{(k)T} d \quad \text{and} \quad f'' = d^T B^{(k)} d.$$

If $f' \geq 0$, go to STEP 2.4.

STEP 2.1 [Find the next breakpoint]

Find the largest stepsize Δt for which $l^{(k)} \leq x + \Delta t\, d \leq u^{(k)}$ and the index set J of the indices of all the variables which first encounter their bounds at $x + \Delta t\, d$.

STEP 2.2 [Test whether the GCP has been found]

If $f'' > 0$ and $0 < -(f'/f'') < \Delta t$, reset $x := x - (f'/f'')\, d$ and go to STEP 2.4.

STEP 2.3 [Update line derivatives]

Compute $b = B^{(k)} \left(\sum_{i \in J} d_i\, e^i\right)$ and reset

$$f' := f' + \Delta t\, f'' - b^T x - \sum_{i \in J} d_i\, g_i,$$

$$f'' := f'' + b^T \left(\sum_{i \in J} d_i\, e^i - 2b\right),$$

$$x := x + \Delta t\, d \quad \text{and}$$

$$d_i := 0 \quad \text{for all} \quad i \in J.$$

If $f' \geq 0$, go to STEP 2.4. Otherwise, go to STEP 2.1.

STEP 2.4 [Termination with GCP]

Set $x^{(k)c} = x$.

STEP 3 [Find the new iterate]

Compute the active set $I(x^{(k)c}, l^{(k)}, u^{(k)})$.

Apply the following variant of the conjugate gradient algorithm to find an approximation $\bar{x}^{(k)}$ to the minimizer of the model function (2.1) over the feasible region (2.2) with the additional restriction that the variables with indices in $I(x^{(k)c}, l^{(k)}, u^{(k)})$ remain fixed at the corresponding values of $x^{(k)c}$:

STEP 3.0 [Initialization]

Set $x = x^{(k)c}$ and $r = -g^{(k)} - B^{(k)}(x - x^{(k)})$. Compute $\eta^{(k)}$ from equation (2.8). Set

$$\hat{I} = \{1, 2, \ldots, n\} \setminus I(x^{(k)c}, l^{(k)}, u^{(k)}).$$

Let $\hat{x}$, $\hat{r}$, $\hat{l}$ and $\hat{u}$ denote the vectors whose components are those of x, r, $l^{(k)}$ and $u^{(k)}$ indexed by $\hat{I}$. Furthermore, let $\hat{B}$ denote the matrix whose rows and columns are those of $B^{(k)}$ indexed by $\hat{I}$. Set $\hat{p} = 0$, $\rho_1 = 1$ and $\rho_2 = \hat{r}^T\hat{r}$.

STEP 3.1 [Test for the required accuracy in c.g. iteration]

If $\rho_2 < (\eta^{(k)})^2$, go to STEP 3.3.

STEP 3.2 [Conjugate gradient recurrences]

Set $\beta = \rho_2 / \rho_1$ and reset $\hat{p} := \hat{r} + \beta\, \hat{p}$. Compute $\hat{y} = \hat{B}\, \hat{p}$ and find α_1, the largest value of α for which $\hat{l} \leq \hat{x} + \alpha\, \hat{p} \leq \hat{u}$.

If $\hat{p}^T\, \hat{y} \leq 0$, reset $\hat{x} := \hat{x} + \alpha_1\, \hat{p}$ and go to STEP 3.3.

Otherwise, calculate $\alpha_2 = \rho_2 / \hat{p}^T\hat{y}$.

If $\alpha_2 > \alpha_1$, reset $\hat{x} := \hat{x} + \alpha_1\, \hat{p}$ and go to STEP 3.3.

Otherwise, reset

$$\hat{x} := \hat{x} + \alpha_2\, \hat{p}$$

$$\hat{r} := \hat{r} - \alpha_2\, \hat{y}$$

$$\rho_1 := \rho_2 \quad \text{and}$$

$$\rho_2 := \hat{r}^T\hat{r}.$$

Return to step 3.1.

STEP 3.3 [Termination of conjugate gradient iteration]

Reset the components of x indexed by $\hat{I}$ to the values of the associated components of $\hat{x}$. Set $\bar{x}^{(k+1)} = x$.

STEP 4 [Compute the ratio of achieved to predicted reduction in the function]

Compute $f(\bar{x}^{(k+1)})$ and

$$\rho^{(k)} = (f(x^{(k)}) - f(\bar{x}^{(k+1)})) / (m^{(k)}(x^{(k)}) - f(\bar{x}^{(k+1)})).$$

STEP 5 [Updates]

Set

$$x^{(k+1)} = \begin{cases} \bar{x}^{(k+1)}) & \text{if } \rho^{(k)} > \mu \\ x^{(k)} & \text{if } \rho^{(k)} \leq \mu, \end{cases}$$

set $\Delta^{(k+1)}$ according to equation (2.4) and compute

$$g^{(k+1)} = \begin{cases} g(x^{(k+1)}) & \text{if } \rho^{(k)} > \mu \\ g^{(k)} & \text{if } \rho^{(k)} \leq \mu. \end{cases}$$

Revise the approximation to the second derivative approximation $B^{(k+1)}$ while ensuring that equation (2.10) is maintained. Increment k by 1 and return to STEP 1.

5. Numerical experiments.

In order to investigate the behaviour of the algorithm stated in §4, we have performed a fairly substantial amount of numerical testing. We have attempted to solve forty six test problems using a variety of methods; the methods only differing in their choice of second derivative information. Our aim is naturally to indicate that our framework is an effective one for solving relatively small problems; our belief is that these tests give some indication as to how our framework could cope with larger problems.

For comparison, we have computed the second derivative matrices $B^{(k)}$ in a number of different ways. We refer the reader to Gill, Murray and Wright (1981, section 4.5.2.1) for a definition of the methods and terminology of this section.

Firstly we have tried using the exact second derivatives; this was intended to allow a comparison of other schemes with the ideal updating formula. Secondly we have included the B.F.G.S. and D.F.P. updating schemes – the update is only performed in either case provided that the new approximation can be ensured to be positive definite. We note that the B.F.G.S. (but not the D.F.P.) update has the property that the updates remain uniformly bounded (see, Powell, 1975, Griewank and Toint, 1982b) for convex problems and hence automatically satisfy the growth requirement (2.10) in this case. Thirdly we have included the P.S.B. update as an example of a scheme which allows indefinite approximations to be generated and for which no effort needs be made to ensure that the growth requirement (2.10) is satisfied. Finally we have tried the Symmetric Rank-one update as an example of an update which allows indefinite approximations but which must be controlled so that the growth requirement (2.10) is satisfied. We choose to skip the update whenever the rank one correction is too large.

For simplicity, we have chosen to skip the update when the correction has norm larger than 10^8.

For each of these second derivative schemes, we solved every test problem twice. For the first run, the problem is essentially unconstrained (U) but we have taken the precaution of including the simple bounds

$$-100.0 \leqslant x_i \leqslant 100.0, \qquad 1 \leqslant i \leqslant n,$$

to prevent an unbounded solution being found. For a few of the problems, typically those containing exponentials, we have provided tighter bounds in order to prevent numerical overflows when the problem functions are evaluated – notice that our framework is especially useful for imposing such safeguards. Details of these additional bounds are given in appendix 1. If x^* denotes the solution obtained for problem U, the second test of the problem (C) includes the additional bounds

$$x_i^* + 0.1 \leqslant x_i \leqslant x_i^* + 1.1 \quad \text{for all odd } i.$$

The starting point for the constrained problem is the projection of the starting point for the unconstrained problem into the feasible region, $P[x^{(0)}]$. Details of the test problems and their solutions are given in appendix 1.

All of the computation was performed on the IBM 3084Q computer at Harwell; our code is written in Double Precision Fortran 77 (with modifications as required by WATFIV) and compiled using the WATFIV Fortran compiler. All timings reported are in seconds for time spent in the C.P.U. and appear to be correct to about one hundredth of a second. As the code is essentially a prototype for studying the effectiveness of the method, we did not feel it necessary to use a more sophisticated optimizing compiler; the timings quoted are supposed to allow comparisons of the relative merits of the schemes tested rather than trying to obtain the "best" timing for a particular scheme. The initial estimate $B^{(0)} = I$ was used for all of the updating schemes; the initial trust region radius $\Delta^{(0)} = 0.1\|\bar{g}^{(0)}\|_2$. We also chose $\mu = 0.25$, $\eta = 0.75$, $\gamma_0 = 0.5$ and $\gamma_2 = 2$.

The results are given in Tables 4.1–4.5 and summarized in Tables 4.6–4.8. For each run we have given the number of variables (n), the number of iterations (it.) required to solve the problem (which is the same as the number of objective function evaluations), the total number of derivative evaluations (de.) (which is the same as the number of iterations for which the trial point is accepted), the total number of conjugate gradient iterations taken (c.g.), the norm of the projected gradient (gr.norm) at the final point, the last iteration on which the active set changes (cas) (which is an indication as to when the correct active set is obtained for successful runs) and the time taken to reach the final point (time). Additionally, for the B.F.G.S., D.F.P. and S.R.1 runs, we have recorded the number of times the second derivative update was skipped (usk). Every run was terminated if the norm of the projected gradient of the objective function was reduced below 10^{-6}. In addition, each unconstrained (constrained) run was terminated if more than max(20n, 600) (max(10n, 300)) iterations were performed or if the trust region radius $\Delta^{(k)}$ was reduced below 10^{-16} and these failures are indicated by $\geqslant$ in the iteration column. In Table 4.6, the performance of the updating schemes is compared with that obtained by the method which uses exact second derivative information. The relative number of iterations and the relative timings are given and indicate how much worse the respective updating schemes are than the use of exact second derivatives. In

Problem	n	it.	de.	c.g.	gr.norm	cas	time
GENROSE U	8	42	31	184	2.2D-07	0	0.88
GENROSE C	8	15	15	54	2.8D-10	6	0.27
CHAINROSE U	25	20	17	98	1.3D-07	0	1.80
CHAINROSE C	25	18	13	26	3.9D-07	14	0.91
DEGENROSE U	25	95	94	34	6.3D-07	95	5.64
DEGENROSE C	25	17	14	13	1.2D-10	12	0.78
GENSING U	20	10	11	37	5.6D-07	0	0.58
GENSING C	20	4	5	3	1.6D-07	0	0.15
CHAINSING U	20	18	19	96	4.6D-07	0	1.21
CHAINSING C	20	3	4	8	2.5D-15	0	0.13
DEGENSING U	20	155	156	56	3.0D-07	148	6.45
DEGENSING C	20	3	4	8	2.5D-15	1	0.13
GENWOOD U	8	107	75	393	2.6D-10	0	2.03
GENWOOD C	8	5	6	2	1.0D-07	0	0.07
CHAINWOOD U	8	77	54	365	2.4D-09	0	1.64
CHAINWOOD C	8	5	6	7	7.8D-07	0	0.07
HOSC45 U	10	19	20	0	0.0D-01	19	0.26
HOSC45 C	10	12	13	0	0.0D-01	12	0.18
BROYDEN1A U	30	11	12	38	6.9D-07	0	1.13
BROYDEN1A C	30	8	9	10	3.5D-10	0	0.54
BROYDEN1B U	30	7	8	31	6.3D-08	0	0.79
BROYDEN1B C	30	6	7	5	3.3D-07	0	0.39
BROYDEN2A U	30	14	15	7	5.6D-07	0	1.48
BROYDEN2A C	30	10	11	11	1.8D-07	3	1.05
BROYDEN2B U	30	9	10	12	7.0D-07	0	1.04
BROYDEN2B C	30	9	10	9	6.5D-08	2	0.93
TOINTBROY U	30	8	9	52	3.3D-07	0	1.10
TOINTBROY C	30	8	9	8	2.2D-10	0	0.55
TRIG U	10	7	6	7	1.6D-07	0	0.28
TRIG C	10	8	8	6	8.8D-15	0	0.32
TOINTTRIG U	10	13	9	26	9.6D-08	0	0.28
TOINTTRIG C	10	10	9	18	2.8D-13	3	0.19
CRAGGLEVY U	8	24	25	106	4.4D-07	0	0.50
CRAGGLEVY C	8	20	20	65	1.7D-07	19	0.36
PENALTY U	15	27	25	58	2.0D-08	0	0.88
PENALTY C	15	80	81	31	1.2D-08	78	2.12
AUGMLAGN U	15	31	21	103	8.1D-07	8	1.15
AUGMLAGN C	15	47	37	186	5.2D-08	44	1.77
BROWN1 U	20	27	28	32	9.4D-10	1	1.22
BROWN1 C	20	27	28	0	3.6D-10	1	0.93
BROWN3 U	20	7	8	4	4.5D-08	0	0.31
BROWN3 C	20	6	7	6	3.9D-07	0	0.24
BVP(N=10) U	10	4	5	31	1.7D-09	0	0.14
BVP(N=10) C	10	4	5	18	1.4D-07	3	0.10
BVP(N=20) U	20	5	6	77	6.4D-10	0	0.63
BVP(N=20) C	20	9	10	62	6.1D-07	8	0.67
VAR(N=20) U	20	6	7	112	2.7D-09	0	0.87
VAR(N=20) C	20	6	7	61	1.7D-09	3	0.53
VAR(N=45) U	45	6	7	234	1.7D-07	0	5.35
VAR(N=45) C	45	12	13	90	1.8D-09	9	3.10

Table 4.1. Results for tests using exact second derivatives

Problem	n	it.	de.	usk	c.g.	gr.norm	cas	time
GENROSE U	8	149	98	0	376	5.9D–07	0	2.82
GENROSE C	8	84	54	0	178	3.0D–08	49	1.33
CHAINROSE U	25	190	129	0	376	6.7D–07	0	15.50
CHAINROSE C	25	65	43	0	97	2.2D–07	16	4.10
DEGENROSE U	25	240	210	0	65	7.6D–07	240	17.39
DEGENROSE C	25	56	28	0	35	6.9D–07	22	2.75
GENSING U	20	125	77	0	503	3.8D–07	0	8.35
GENSING C	20	15	10	0	10	1.9D–07	12	0.58
CHAINSING U	20	165	109	0	512	6.3D–07	0	10.44
CHAINSING C	20	25	15	0	14	2.0D–08	0	0.93
DEGENSING U	20	572	545	0	203	4.4D–07	572	29.74
DEGENSING C	20	21	13	0	16	1.1D–07	7	0.77
GENWOOD U	8	201	136	0	720	6.5D–08	0	3.99
GENWOOD C	8	50	23	0	27	1.2D–07	21	0.59
CHAINWOOD U	8	221	148	0	894	4.9D–07	0	4.67
CHAINWOOD C	8	47	25	0	40	2.4D–08	22	0.61
HOSC45 U	10	>600	601	600	0	2.3D–04	0	7.57
HOSC45 C	10	86	87	86	0	0.0D–01	86	1.07
BROYDEN1A U	30	65	44	0	218	8.4D–07	0	7.68
BROYDEN1A C	30	54	23	0	17	4.1D–07	13	3.70
BROYDEN1B U	30	97	62	0	44	5.4D–07	0	8.78
BROYDEN1B C	30	65	31	0	30	3.3D–07	0	4.46
BROYDEN2A U	30	71	36	0	183	8.8D–07	0	7.66
BROYDEN2A C	30	65	29	0	18	1.9D–07	18	5.15
BROYDEN2B U	30	69	34	0	14	2.3D–07	0	6.29
BROYDEN2B C	30	67	29	0	25	4.2D–07	12	5.06
TOINTBROY U	30	122	80	0	76	7.5D–07	0	11.54
TOINTBROY C	30	68	34	0	29	3.8D–07	10	4.89
TRIG U	10	45	21	0	46	5.4D–07	0	1.09
TRIG C	10	17	11	0	13	3.3D–07	0	0.37
TOINTTRIG U	10	73	37	0	65	9.3D–07	0	1.50
TOINTTRIG C	10	57	25	0	51	1.3D–07	7	0.86
CRAGGLEVY U	8	99	83	0	574	6.3D–07	0	2.38
CRAGGLEVY C	8	63	56	0	151	7.6D–07	56	1.10
PENALTY U	15	139	104	2	537	7.0D–07	0	6.28
PENALTY C	15	74	59	0	156	1.7D–07	61	2.63
AUGMLAGN U	15	156	112	1	355	4.5D–07	41	5.83
AUGMLAGN C	15	139	98	0	306	9.7D–07	131	5.03
BROWN1 U	20	94	54	3	159	2.9D–07	8	4.68
BROWN1 C	20	33	29	3	0	9.0D–07	6	1.21
BROWN3 U	20	18	15	0	28	5.1D–07	0	0.94
BROWN3 C	20	9	9	0	13	7.3D–07	0	0.41
BVP(N=10) U	10	31	27	0	109	6.1D–07	0	0.80
BVP(N=10) C	10	33	23	0	82	2.4D–08	20	0.70
BVP(N=20) U	20	69	56	0	438	6.1D–07	0	5.67
BVP(N=20) C	20	61	47	0	277	1.8D–07	37	4.09
VAR(N=20) U	20	134	97	0	390	1.7D–07	0	8.35
VAR(N=20) C	20	128	86	0	267	1.9D–07	88	7.09
VAR(N=45) U	45	366	259	0	1252	5.9D–07	0	85.94
VAR(N=45) C	45	337	234	0	772	8.3D–07	279	69.54

Table 4.2. Results for tests using the B.F.G.S. update

Problem	n	it.	de.	usk	c.g.	gr.norm	cas	time
GENROSE U	8	>600	589	0	1154	4.7D 00	0	10.96
GENROSE C	8	39	31	0	99	5.4D-07	18	0.67
CHAINROSE U	25	95	70	0	410	7.1D-07	0	9.39
CHAINROSE C	25	45	30	0	47	4.0D-07	11	2.80
DEGENROSE U	25	183	164	0	202	9.4D-07	183	14.18
DEGENROSE C	25	47	20	0	43	2.6D-07	10	2.27
GENSING U	20	76	61	0	338	9.4D-07	0	5.30
GENSING C	20	14	10	0	8	3.0D-07	12	0.58
CHAINSING U	20	253	230	0	1635	4.4D-07	0	21.04
CHAINSING C	20	17	12	0	13	9.6D-07	0	0.69
DEGENSING U	20	>600	594	0	9	8.0D-03	601	32.08
DEGENSING C	20	19	12	0	13	1.7D-09	7	0.71
GENWOOD U	8	>600	587	92	1824	1.9D 00	0	11.46
GENWOOD C	8	28	16	0	10	1.7D-08	16	0.34
CHAINWOOD U	8	>600	590	0	1904	1.4D-01	0	12.10
CHAINWOOD C	8	30	18	0	24	1.7D-08	24	0.42
HOSC45 U	10	>600	601	600	0	2.3D-04	0	7.55
HOSC45 C	10	86	87	86	0	0.0D-01	86	1.07
BROYDEN1A U	30	96	69	0	819	9.5D-07	0	16.51
BROYDEN1A C	30	59	24	0	52	1.8D-07	13	4.11
BROYDEN1B U	30	65	36	0	62	6.0D-07	0	5.91
BROYDEN1B C	30	52	20	0	44	3.4D-07	0	3.49
BROYDEN2A U	30	83	51	0	432	9.3D-07	0	11.17
BROYDEN2A C	30	69	32	0	124	9.3D-08	14	5.69
BROYDEN2B U	30	66	26	0	69	4.9D-07	0	5.75
BROYDEN2B C	30	65	29	0	95	9.7D-07	12	5.07
TOINTBROY U	30	77	44	0	108	9.9D-07	0	7.41
TOINTBROY C	30	56	26	0	53	7.7D-07	10	4.03
TRIG U	10	28	16	0	20	6.1D-07	0	0.71
TRIG C	10	15	10	0	10	1.2D-07	3	0.34
TOINTTRIG U	10	41	24	0	71	1.7D-07	0	0.91
TOINTTRIG C	10	26	15	0	30	4.0D-07	7	0.44
CRAGGLEVY U	8	>600	592	0	2387	1.1D-04	0	12.70
CRAGGLEVY C	8	193	189	0	237	9.3D-07	170	3.06
PENALTY U	15	>600	572	1	2971	8.6D-01	0	30.19
PENALTY C	15	157	142	0	588	8.3D-07	63	5.78
AUGMLAGN U	15	>600	586	0	1346	3.6D-01	395	24.73
AUGMLAGN C	15	>300	291	0	998	2.5D-01	297	12.12
BROWN1 U	20	>600	565	3	2717	1.3D-04	8	41.86
BROWN1 C	20	33	29	3	0	9.0D-07	6	1.19
BROWN3 U	20	15	12	0	18	8.1D-07	0	0.75
BROWN3 C	20	9	9	0	11	7.0D-07	0	0.41
BVP(N=10) U	10	47	45	0	252	3.6D-08	0	1.41
BVP(N=10) C	10	25	22	0	83	1.4D-07	12	0.59
BVP(N=20) U	20	269	264	0	2645	8.5D-07	0	27.07
BVP(N=20) C	20	107	97	0	734	2.7D-07	30	8.46
VAR(N=20) U	20	56	43	0	226	1.6D-07	0	3.82
VAR(N=20) C	20	54	40	0	185	8.6D-08	21	3.40
VAR(N=45) U	45	130	99	0	695	1.5D-07	0	35.54
VAR(N=45) C	45	153	127	0	600	9.3D-07	136	36.88

Table 4.3. Results for tests using the D.F.P. update

Problem	n	it.	de.	c.g.	gr.norm	cas	time
GENROSE U	8	177	125	581	2.2D-07	0	3.62
GENROSE C	8	96	67	196	2.0D-07	42	1.52
CHAINROSE U	25	137	96	323	8.1D-07	0	12.21
CHAINROSE C	25	52	36	75	3.9D-07	12	3.17
DEGENROSE U	25	233	212	23	6.4D-07	233	17.08
DEGENROSE C	25	42	20	39	8.1D-08	10	2.01
GENSING U	20	171	132	387	8.4D-07	0	10.21
GENSING C	20	14	9	7	8.7D-07	9	0.51
CHAINSING U	20	217	173	1745	5.1D-07	0	19.91
CHAINSING C	20	24	14	21	5.2D-07	0	0.88
DEGENSING U	20	>600	559	547	5.4D-05	601	31.50
DEGENSING C	20	19	12	12	2.4D-09	7	0.68
GENWOOD U	8	296	230	1143	5.7D-07	0	6.19
GENWOOD C	8	32	15	19	2.8D-08	13	0.36
CHAINWOOD U	8	279	204	1099	9.2D-07	0	5.97
CHAINWOOD C	8	31	20	21	5.4D-09	23	0.44
HOSC45 U	10	29	30	0	0.0D-01	29	0.50
HOSC45 C	10	15	16	4	0.0D-01	15	0.28
BROYDEN1A U	30	172	143	1905	9.9D-07	0	34.55
BROYDEN1A C	30	60	28	58	5.7D-07	13	4.10
BROYDEN1B U	30	87	50	96	8.8D-07	0	8.23
BROYDEN1B C	30	57	23	42	1.1D-07	0	3.67
BROYDEN2A U	30	111	68	595	9.8D-07	0	15.22
BROYDEN2A C	30	69	37	60	7.4D-07	18	5.44
BROYDEN2B U	30	68	28	48	3.4D-07	0	5.88
BROYDEN2B C	30	62	26	69	7.1D-07	16	4.68
TOINTBROY U	30	99	59	112	5.9D-07	0	9.72
TOINTBROY C	30	54	23	28	9.7D-07	14	3.74
TRIG U	10	31	19	27	3.3D-07	0	0.79
TRIG C	10	16	10	11	9.6D-08	3	0.35
TOINTTRIG U	10	35	23	45	6.0D-08	0	0.78
TOINTTRIG C	10	22	14	21	3.3D-07	7	0.38
CRAGGLEVY U	8	454	422	2967	9.3D-07	0	11.82
CRAGGLEVY C	8	250	240	651	1.0D-06	186	4.14
PENALTY U	15	566	514	2886	9.6D-07	0	28.96
PENALTY C	15	>300	261	159	1.1D 00	109	8.96
AUGMLAGN U	15	549	493	4498	1.0D-06	24	33.26
AUGMLAGN C	15	127	103	315	8.1D-08	116	4.77
BROWN1 U	20	>600	539	7570	8.6D-04	528	69.40
BROWN1 C	20	41	37	0	5.3D-07	6	1.47
BROWN3 U	20	21	13	29	3.4D-08	0	0.99
BROWN3 C	20	9	9	13	3.8D-07	0	0.40
BVP(N=10) U	10	34	26	114	6.5D-07	0	0.86
BVP(N=10) C	10	38	27	92	4.9D-07	19	0.81
BVP(N=20) U	20	113	82	983	5.6D-07	0	10.62
BVP(N=20) C	20	131	88	867	1.6D-07	17	9.99
VAR(N=20) U	20	119	85	470	5.5D-07	0	8.15
VAR(N=20) C	20	102	72	353	9.3D-07	74	6.40
VAR(N=45) U	45	271	194	1390	7.7D-07	0	77.81
VAR(N=45) C	45	200	157	619	1.1D-07	168	47.60

Table 4.4. Results for tests using the P.S.B. update

Problem	n	it.	de.	usk	c.g.	gr.norm	cas	time
GENROSE U	8	195	115	0	467	4.8D-07	0	3.39
GENROSE C	8	70	37	0	118	8.8D-09	32	0.96
CHAINROSE U	25	140	77	0	220	1.6D-07	0	10.47
CHAINROSE C	25	39	23	0	55	4.0D-07	11	2.24
DEGENROSE U	25	152	120	0	58	9.2D-08	152	10.17
DEGENROSE C	25	34	16	0	22	1.0D-08	10	1.59
GENSING U	20	74	46	0	247	6.0D-07	0	4.49
GENSING C	20	12	8	0	7	2.3D-10	0	0.43
CHAINSING U	20	83	58	0	445	8.7D-07	0	6.09
CHAINSING C	20	14	9	0	9	7.1D-10	0	0.50
DEGENSING U	20	>600	570	0	539	9.4D-04	77	28.69
DEGENSING C	20	14	9	0	8	5.1D-09	7	0.48
GENWOOD U	8	486	307	0	814	6.8D-07	0	7.72
GENWOOD C	8	32	15	0	4	1.0D-07	28	0.37
CHAINWOOD U	8	411	261	0	1148	1.1D-10	0	7.57
CHAINWOOD C	8	23	15	0	15	1.4D-09	16	0.29
HOSC45 U	10	28	29	0	0	0.0D-01	28	0.45
HOSC45 C	10	14	15	0	2	0.0D-01	14	0.25
BROYDEN1A U	30	129	76	0	704	9.5D-07	0	17.46
BROYDEN1A C	30	54	25	0	46	4.2D-08	13	3.63
BROYDEN1B U	30	81	40	0	78	2.2D-07	0	7.33
BROYDEN1B C	30	45	19	0	45	3.0D-07	0	2.82
BROYDEN2A U	30	95	52	0	284	4.0D-07	0	10.47
BROYDEN2A C	30	63	30	0	44	5.3D-07	25	5.04
BROYDEN2B U	30	82	38	0	71	1.5D-07	0	7.68
BROYDEN2B C	30	57	28	0	45	9.3D-07	12	4.35
TOINTBROY U	30	62	36	0	119	4.2D-07	0	6.10
TOINTBROY C	30	44	22	0	35	6.2D-07	19	3.25
TRIG U	10	22	13	0	25	1.6D-08	0	0.55
TRIG C	10	13	9	0	9	3.3D-09	0	0.28
TOINTTRIG U	10	27	17	0	30	5.8D-07	0	0.57
TOINTTRIG C	10	20	12	0	19	1.9D-10	7	0.33
CRAGGLEVY U	8	142	92	0	398	2.6D-07	0	2.54
CRAGGLEVY C	8	56	48	0	182	3.2D-07	49	1.02
PENALTY U	15	163	105	0	742	5.2D-08	0	7.45
PENALTY C	15	91	71	0	126	4.9D-07	81	2.81
AUGMLAGN U	15	125	92	0	313	4.4D-07	37	4.47
AUGMLAGN C	15	97	72	0	240	2.1D-07	90	3.39
BROWN1 U	20	110	88	4	164	3.4D-07	8	5.41
BROWN1 C	20	33	29	3	0	9.0D-07	6	1.08
BROWN3 U	20	12	11	0	10	4.4D-09	0	0.54
BROWN3 C	20	9	9	0	12	1.8D-07	0	0.37
BVP(N=10) U	10	27	19	0	42	2.1D-08	0	0.54
BVP(N=10) C	10	18	13	0	45	4.9D-08	12	0.36
BVP(N=20) U	20	29	25	0	143	2.5D-07	0	2.01
BVP(N=20) C	20	35	26	0	172	2.6D-07	24	2.26
VAR(N=20) U	20	41	32	0	146	2.8D-08	0	2.52
VAR(N=20) C	20	35	25	0	92	9.4D-08	19	1.92
VAR(N=45) U	45	78	59	0	457	1.6D-08	0	22.21
VAR(N=45) C	45	85	67	0	245	9.2D-08	80	17.98

Table 4.5. Results for tests using the symmetric rank one update

Problem	BFGS it.	BFGS time	DFP it.	DFP time	PSB it.	PSB time	SR1 it.	SR1 time
GENROSE U	3.55	3.14	>14.29	>12.12	4.21	4.01	4.64	3.75
GENROSE C	5.60	4.87	2.60	2.46	6.40	5.55	4.67	3.52
CHAINROSE U	9.50	8.49	4.75	5.15	6.85	6.69	7.00	5.74
CHAINROSE C	3.61	4.49	2.50	3.06	2.89	3.47	2.17	2.45
DEGENROSE U	2.53	3.08	1.93	2.51	2.45	3.02	1.60	1.80
DEGENROSE C	3.29	3.51	2.76	2.90	2.47	2.57	2.00	2.04
GENSING U	12.50	13.78	7.60	8.75	17.10	16.83	7.40	7.42
GENSING C	3.75	3.82	3.50	3.81	3.50	3.34	3.00	2.80
CHAINSING U	9.17	8.48	14.06	17.08	12.06	16.16	4.61	4.95
CHAINSING C	8.33	6.63	5.67	4.92	8.00	6.32	4.67	3.59
DEGENSING U	3.69	4.59	> 3.87	> 4.95	> 3.87	> 4.86	> 3.87	> 4.43
DEGENSING C	7.00	5.52	6.33	5.06	6.33	4.84	4.67	3.45
GENWOOD U	1.88	1.95	> 5.61	> 5.59	2.77	3.02	4.54	3.76
GENWOOD C	10.00	8.78	5.60	5.04	6.40	5.37	6.40	5.54
CHAINWOOD U	2.87	2.83	> 7.79	> 7.31	3.62	3.61	5.34	4.57
CHAINWOOD C	9.40	8.30	6.00	5.71	6.20	5.99	4.60	4.03
HOSC45 U	>31.58	>26.58	>31.58	>26.52	1.53	1.79	1.47	1.62
HOSC45 C	7.17	5.83	7.17	5.87	1.25	1.57	1.17	1.36
BROYDEN1A U	5.91	6.62	8.73	14.22	15.64	29.74	11.73	15.04
BROYDEN1A C	6.75	6.74	7.38	7.49	7.50	7.48	6.75	6.62
BROYDEN1B U	13.86	10.70	9.29	7.21	12.43	10.02	11.57	8.93
BROYDEN1B C	10.83	11.06	8.67	8.66	9.50	9.12	7.50	7.00
BROYDEN2A U	5.07	5.08	5.93	7.40	7.93	10.08	6.79	6.94
BROYDEN2A C	6.50	4.84	6.90	5.35	6.90	5.12	6.30	4.74
BROYDEN2B U	7.67	5.92	7.33	5.41	7.56	5.53	9.11	7.22
BROYDEN2B C	7.44	5.40	7.22	5.42	6.89	5.00	6.33	4.65
TOINTBROY U	15.25	10.27	9.63	6.59	12.38	8.65	7.75	5.44
TOINTBROY C	8.50	8.68	7.00	7.16	6.75	6.64	5.50	5.78
TRIG U	6.43	3.65	4.00	2.38	4.43	2.64	3.14	1.86
TRIG C	2.13	1.15	1.88	1.07	2.00	1.08	1.63	0.88
TOINTTRIG U	5.62	5.09	3.15	3.10	2.69	2.67	2.08	1.97
TOINTTRIG C	5.70	4.54	2.60	2.32	2.20	1.99	2.00	1.75
CRAGGLEVY U	4.13	4.59	>25.00	>24.36	18.92	22.68	5.92	4.89
CRAGGLEVY C	3.15	3.03	9.65	8.41	12.50	11.38	2.80	2.80
PENALTY U	5.15	6.94	>22.22	>33.34	20.96	31.98	6.04	8.24
PENALTY C	0.92	1.24	1.96	2.72	> 3.75	> 4.21	1.14	1.32
AUGMLAGN U	5.03	4.97	>19.35	>21.04	17.71	28.28	4.03	3.81
AUGMLAGN C	2.96	2.83	> 6.38	> 6.82	2.70	2.68	2.06	1.91
BROWN1 U	3.48	3.79	>22.22	>33.74	>22.22	>55.94	4.07	4.37
BROWN1 C	1.22	1.30	1.22	1.28	1.52	1.58	1.22	1.16
BROWN3 U	2.57	2.89	2.14	2.32	3.00	3.05	1.71	1.68
BROWN3 C	1.50	1.71	1.50	1.70	1.50	1.63	1.50	1.54
BVP(N=10) U	7.75	4.98	11.75	8.70	8.50	5.30	6.75	3.39
BVP(N=10) C	8.25	6.53	6.25	5.50	9.50	7.50	4.50	3.41
BVP(N=20) U	13.80	8.64	53.80	41.16	22.60	16.16	5.80	3.07
BVP(N=20) C	6.78	6.10	11.89	12.61	14.56	14.89	3.89	3.37
VAR(N=20) U	22.33	9.38	9.33	4.30	19.83	9.16	6.83	2.84
VAR(N=20) C	21.33	13.13	9.00	6.30	17.00	11.86	5.83	3.57
VAR(N=45) U	61.00	16.06	21.67	6.70	45.17	13.94	13.00	3.98
VAR(N=45) C	28.08	22.83	12.75	11.95	16.67	14.70	7.08	5.56

Table 4.6. A comparison of the performance of the methods relative to the method using exact second derivatives

Ranking	Exact	B.F.G.S.	D.F.P.	P.S.B.	S.R.1
1st	49	1	0	0	0
2nd	1	12	7	1	35
3rd	0	7	19	16	7
4th	0	7	11	20	6
5th	0	22	3	10	1
Failure	0	1	10	3	1

Table 4.7. A ranking of the various update methods according to the number of iterations taken

Ranking	Exact	B.F.G.S.	D.F.P.	P.S.B.	S.R.1
1st	49	0	0	0	1
2nd	1	10	5	0	34
3rd	0	8	17	15	9
4th	0	9	11	23	4
5th	0	22	7	9	1
Failure	0	1	10	3	1

Table 4.8. A ranking of the various update methods according to the time taken

Tables 4.7 and 4.8, the methods are ranked according to the numbers of iterations and the c.p.u. times taken for each test problem. For each problem, the five methods are ranked from 1 (fewest iterations or shortest time) to 5 (most iterations or longest time) with failures being recorded separately. The table entries give the sums of these rankings over all the test problems. For instance, the method using the D.F.P. update ranks third for 19 of the test problems when the ranking is based upon the number of iterations taken.

As one might expect, the use of exact second derivatives gives uniformly and significantly better results than the other approximating schemes – both in terms of the number of iterations and in terms of the overall timings. Of course, none of the problems solved has particularly complicated derivatives and the calculation of exact second

derivatives is often cheaper than performing a rank one or two update to an existing approximation. Nonetheless, we believe that the use of second derivatives in our framework is worthwhile and would recommend their use whenever they are available. The worst performance occurs on the problem DEGENSING for which the complementary slackness condition does not hold at the solution. A closer examination of the run shows that the degenerate bounds repeatedly enter and leave the active set as the solution is approached (as is to be expected). The insistence on terminating the conjugate gradient (c.g.) iteration when a previously inactive variable hits a bound slows down convergence in the presence of degenerate bounds as it forces small steps to be taken. As stated in §3, the reasoning behind this termination criterion for the c.g. iteration is that for non-degenerate problems the correct active set will eventually be found at the GCP and the c.g. iteration will thereafter be unhindered by the inactive bounds. As the c.g. scheme would have to be restarted every time a new bound is encountered (in order to restore conjugacy of the c.g. search directions) and as this could involve a lot of additional computation for large scale problems, our termination criteria seemed reasonable. However, for this particular problem, the inactive degenerate bounds were all encountered during the *first* c.g. iteration and there would be little overhead incurred in restarting the iteration under these conditions. The convergence theory of Conn, Gould and Toint (1986) allows us to increase the active set in the c.g. iteration provided that it always contains those variables active at the GCP. In Table 4.9, we indicate the improvements possible if we allow restarting within the conjugate gradient iteration for the DEGENSING (U) problem.

Hessian	it.	de.	usk	c.g.	gr.norm	cas	time
Exact	20	21	–	142	9.3D–07	9	2.68
B.F.G.S.	104	78	0	531	5.2D–07	86	11.50
D.F.P	203	189	0	1203	7.5D–07	203	23.77
P.S.B.	160	127	–	1033	6.8D–07	160	17.69
S.R.1	85	64	0	459	4.2D–07	64	9.49

Table 4.9. Results for the DEGENSING (U, $n = 20$) problem with restarts allowed in the conjugate gradient iteration

Somewhat more surprisingly, the simplest of the updating schemes, the symmetric rank-one method, appears to perform the best of the other methods in our tests. The use of a trust region removes the main disadvantage of such methods in allowing a meaningful step to be taken even when the approximation is indefinite. For large problems, it is desirable to allow indefinite approximations, as the combination of symmetric secant updating, positive definiteness and preservation of the sparsity structure can lead to severe numerical difficulties (see, Sorensen, 1981). Of course such a scheme must be used with care, but the restriction (2.10) seems to provide a useful

stabilizing effect – we note, however, that the update was almost never skipped. We suspect that the success of the method is due in part to our observation that, in contrast to the other updating schemes, the second derivative approximations with respect to the inactive variables at the solution converged to their true values on every problem tested. (Such a result has been theoretically established, under very mild conditions, for quadratic functions (see, e.g. Fletcher (1980, p.41) and would appear to be true in general.) This is important for trust region methods which are essentially based on being able to model the true function as accurately as possible in a larger subspace than just that given by the Newton direction. The use of S.R.1 for larger problems is thus quite appealing.

All of the updating methods failed on at least one problem. Of the three other methods the B.F.G.S. update appears to be the most reliable – indeed the only failure for this problem, on HOSC45, is attributable to our insistence on maintaining a positive definite approximation to a matrix which is uniformly indefinite. As has been observed in the past, the B.F.G.S. method is more robust then the D.F.P. However, in comparison with S.R.1, the performance of B.F.G.S. is surprisingly disappointing although it does perform better on some problems. Indeed, the number of problems for which this update performs worst of all is alarmingly high. The P.S.B. update appears to give more reliable results than the D.F.P. However, it seems somewhat surprising, in view of the relative success of the S.R.1 formula and our belief that part of this success is attributable to the convergence of the updates to the true second derivatives, that the P.S.B. formula performs so poorly. Under the same weak conditions that are needed to prove the convergence of the updates in the S.R.1 case, the P.S.B. method should also generate convergent second derivative approximations (see, Powell, 1970) but, in practice, this convergence often seems to be very slow. The D.F.P. update is the most unreliable. It is interesting to observe that it often fails or performs poorly on problems for which B.F.G.S. is efficient.

A number of different initial approximations $B^{(0)}$ have also been tried, including the use of the exact second derivatives (modified to be positive definite if necessary) and the suggestions of Shanno and Phua (1978) and Dennis and Schnabel (1983). As was to be expected, such different initial approximations did not significantly alter the numerical results, since the problems cited are all reasonably well scaled.

6. Discussion.

We feel that our results indicate that our framework is a good one in which to consider solving problem (1.1). The issues involved in solving larger problems are more complicated. We would not, for instance, imagine storing dense matrices. Our choice of a conjugate gradient " inner iteration " does not require we access matrices, but merely that we are able to form matrix-vector products – the choice of algorithm has always had the large-scale case in mind. To this end we envisage using the partial separability of the problem functions (see, Griewank and Toint, 1982a) to allow efficient storage and updating of matrices in matrix-vector product form. Of course it is essential to use preconditioning when attempting to solve such large problems and we are currently experimenting with a number of preconditioners. The theory given in our previous paper (Conn, Gould and Toint, 1986) has this in mind.

The final aim of this research is actually to produce effective methods for solving

general nonlinear programming problems. Our intention here is to solve problems of this form by combining the nonlinear constraints, in a suitable fashion, with the objective function (for instance, an augmented Lagrangian function) and solving the resulting (sequence of) bound constrained minimization problem(s) using the methods described in this paper. We anticipate there being an interesting tradeoff between the accuracy required in solving the sequence of bound constrained problems and the convergence of the overall method.

7. Appendix 1.

In our tests, we have attempted to solve the following test problems. For each problem, we give (a) the function $f(x)$, (b) any bounds on the variables for the "unconstrained" problem, (c) the starting point $x^{(0)}$ and (d) and (e) the optimal solutions x_C^* and x_U^* obtained for the unconstrained and the constrained problems.

1. The Generalized Rosenbrock function (GENROSE) (Moré, Garbow and Hillstrom, 1981)

(a) $f(x) = 1 + \sum_{i=2}^{n} [100(x_i - x_{i-1}^2)^2 + (1 - x_{i-1})^2]$.

(c) $x^{(0)} = (-1.2, 1, -1.2, 1, 1, 1, \ldots, 1, 1)$.

(d) $x_U^* = (1, 1, 1, \ldots, 1)$.

(e) $x_C^* = (1.1, 1.0775, 1.1, 1.0972, 1.1528, 1.3075, 1.7026, 2.8987)$ $(n = 8)$.

2. The Chained Rosenbrock function (CHAINROSE) (Toint, 1978)

(a) $f(x) = 1 + \sum_{i=2}^{n} [4\alpha_i(x_i - x_{i-1}^2)^2 + (1 - x_{i-1})^2]$,

where the constants α_i are as given by Toint(1978).

(c) $x^{(0)} = (-1, -1, -1, \ldots, -1)$.

(d) $x_U^* = (1, 1, 1, \ldots, 1)$.

(e) $x_C^* = (1.1, 1.0659, 1.1, 1.0711, 1.1, 1.0645, 1.1, 1.0788, 1.1,$
$1.0691, 1.1, 1.0811, 1.1, 1.0759, 1.1, 1.0720, 1.1, 1.0714,$
$1.1, 1.0684, 1.1, 1.0652, 1.1, 1.1782, 1.3881)$ $(n = 25)$.

3. The Degenerate Chained Rosenbrock function (DEGENROSE)

The same as 2 except that

(b) $x_i \leqslant 1$ for all i such that $i \bmod 3 = 0$.

(e) $x_C^* = (1.1, 1.0659, 1.1, 1.0711, 1.1, 1, 1.1, 1.0788, 1.1, 1.0691,$
$1.1, 1, 1.1, 1.0759, 1.1, 1.0720, 1.1, 1, 1.1, 1.0684, 1.1,$
$1.0652, 1.1, 1, 1.3881)$ $(n = 25)$.

4. The Generalized Singular function (GENSING) (Moré, Garbow and Hillstrom, 1981)

(a) $f(x) = \sum_{i \in J} [(x_i + 10x_{i+1})^2 + 5(x_{i+2} - x_{i+3})^2 + (x_{i+1} - 2x_{i+2})^4$
$+ 10(x_i - 10x_{i+3})^4]$,

where n is a multiple of 4 and $J = \{1, 5, 9, \ldots, n-3\}$.

(c) $x^{(0)} = (3, -1, 0, 1, 3, -1, 0, 1, \ldots, 3, -1, 0, 1)$,

(d) $x_U^* = (0, 0, 0, \ldots, 0)$.

(e) $x_C^* = (0.1, \sigma, 0.1, 0.1, \ldots, 0.1, \sigma, 0.1, 0.1)$,

where $\sigma = -9.8153D{-}3$ $(n = 20)$.

5. The Chained Singular function (CHAINSING)

(a) $f(x) = \sum_{i \in J}[(x_i + 10x_{i+1})^2 + 5(x_{i+2} - x_{i+3})^2 + (x_{i+1} - 2x_{i+2})^4 + 10(x_i - 10x_{i+3})^4]$,

where n is a multiple of 4 and $J = \{1, 3, 5, \ldots, n-3\}$.

(c) $x^{(0)} = (3, -1, 0, 1, 3, -1, 0, 1, \ldots, 3, -1, 0, 1)$,

(d) $x_U^* = (0, 0, 0, \ldots, 0)$,

(e) $x_C^* = (0.1, \sigma, 0.1, \omega, 0.1, \omega, 0.1, \omega, 0.1, \omega, 0.1, \omega, 0.1, \omega, 0.1, 0.1)$,

where $\sigma = -9.8153D{-}3$ and $\omega = -4.3827D{-}3$ $(n = 20)$.

6. The Degenerate Chained Singular function (DEGENSING)

The same as 5 except that

(b) $x_i \leq 0$ for all i such that $i \bmod 3 = 0$ and $i \bmod 4 = 2$ and $x_i \geq 0$ for all i such that $i \bmod 3 = 0$ and $i \bmod 4 \neq 2$.

(e) $x_C^* = (0.1, \sigma, 0.1, \omega, 0.1, \omega, 0.1, \omega, 0.1, \omega, 0.1, 0, 0.1, \omega, 0.1, \omega, 0.1, \omega, 0.1, 0.1)$,

where $\sigma = -9.8153D{-}3$ and $\omega = -4.3827D{-}3$ $(n = 20)$.

7. The Generalized Wood function (GENWOOD) (see, Moré, Garbow and Hillstrom, 1981, for the Wood function)

(a) $f(x) = 1 + \sum_{i \in J}[(100(x_{i+1} - x_i^2)^2 + (1 - x_i)^2 + 90(x_{i+3} - x_{i+2}^2)^2 + (1 - x_{i+2})^2 + 10(x_{i+1} + x_{i+3} - 2)^2 + 0.1(x_{i+1} - x_{i+3})^2]$,

where n is a multiple of 4 and $J = \{1, 5, 9, \ldots, n-3\}$.

(c) $x^{(0)} = (-3, -1, -3, -1, -2, 0, -2, 0, \ldots, -2, 0)$.

(d) $x_U^* = (1, 1, 1, \ldots, 1)$.

(e) $x_C^* = (1.1, 1.1753, 1.1, 1.1715, \ldots, 1.1, 1.1753, 1.1, 1.1715)$.

8. The Chained Wood function (CHAINWOOD)

(a) $f(x) = 1 + \sum_{i \in J}[(100(x_{i+1} - x_i^2)^2 + (1 - x_i)^2 + 90(x_{i+3} - x_{i+2}^2)^2 + (1 - x_{i+2})^2 + 10(x_{i+1} + x_{i+3} - 2)^2 + 0.1(x_{i+1} - x_{i+3})^2]$,

where n is a multiple of 4 and $J = \{1, 3, 5, \ldots, n-3\}$.

(c) $x^{(0)} = (-3, -1, -3, -1, -2, 0, -2, 0, \ldots, -2, 0)$.

(d) $x_U^* = (1, 1, 1, \ldots, 1)$.

(e) $x_C^* = (1.1, 1.1751, 1.1, 1.1734, 1.1, 1.1736, 1.1, 1.1716)(n = 8)$.

9. A generalization of Hock and Schittkowski's 45th problem (HOSC45) (Hock and Schittkowski, 1981)

(a) $f(x) = 2 - \prod_{i=1}^{n} x_i / n!$.

(b) $0 \leq x_i \leq i$ for $1 \leq i \leq n$.

(c) $x^{(0)} = (2, 2, 2, \ldots, 2)$.

(d) $x_U^* = (1, 2, 3, \ldots, n)$.

(e) $x_C^* = (2.1, 2, 4.1, 4, 6.1, 6, \ldots)$.

10. A generalization of the Broyden Tridiagonal function (BROYDEN1A) (see Moré, Garbow and Hillstrom, 1981, for the Broyden Tridiagonal function)

(a) $f(x) = 1 + \sum_{i=1}^{n} |(3 - 2x_i)x_i - x_{i-1} - x_{i+1} + 1|^p$,

where $p = 7/3$ and $x_0 = x_{n+1} = 0$.

(c) $x^{(0)} = (-1, -1, -1, \ldots, -1)$.

(d) $x_U^* = (-0.5707, -0.6819, -0.7025, -0.7063, -0.7070, -0.7071, -0.7071,$
$-0.7071, -0.7071, -0.7071, -0.7071, -0.7071, -0.7071, -0.7071,$
$-0.7071, -0.7071, -0.7071, -0.7071, -0.7071, -0.7071, -0.7071,$
$-0.7070, -0.7068, -0.7064, -0.7051, -0.7015, -0.6919, -0.6658,$
$-0.5960, -0.4164)$, $(n = 30)$.

(e) $x_C^* = (-0.4707, -0.5909, -0.6025, -0.6196, -0.6070, -0.6200, -0.6071,$
$-0.6200, -0.6071, -0.6200, -0.6071, -0.6200, -0.6017, -0.6200,$
$-0.6071, -0.6200, -0.6071, -0.6200, -0.6017, -0.6200, -0.6071,$
$-0.6200, -0.6068, -0.6193, -0.6050, -0.6146, -0.5919, -0.5758,$
$-0.4960, -0.3570)$, $(n = 30)$.

11. Another generalization of the Broyden Tridiagonal function (BROYDEN1B)

The same as 10 except that $p = 2$ and

(e) $x_C^* = (-0.4707, -0.5952, -0.6025, -0.6233, -0.6070, -0.6239, -0.6071,$
$-0.6239, -0.6071, -0.6239, -0.6071, -0.6239, -0.6017, -0.6239,$
$-0.6071, -0.6239, -0.6071, -0.6239, -0.6017, -0.6239, -0.6071,$
$-0.6238, -0.6068, -0.6232, -0.6050, -0.6183, -0.5919, -0.5794,$
$-0.4960, -0.3625)$, $(n = 30)$.

12. A generalization of the Broyden Banded function (BROYDEN2A) (see Moré, Garbow and Hillstrom, 1981, for the Broyden Banded function)

(a) $f(x) = 1 + \sum_{i=1}^{n} |(2 + 5x_i^2)x_i + 1 + \sum_{j \in J_i} x_j(1 + x_j)|^p$,

where $p = 7/3$ and $J_i = \{j: \max(1, i - 5) \leq \min(n, i + 1)\}$.

(c) $x^{(0)} = (-1, -1, -1, \ldots, -1)$.

(d) $x_U^* = (-0.4774, -0.5204, -0.5584, -0.5921, -0.6223, -0.6505, -0.6481,$
$-0.6456, -0.6436, -0.6422, -0.6415, -0.6418, -0.6420, -0.6422,$
$-0.6422, -0.6422, -0.6422, -0.6422, -0.6422, -0.6422, -0.6422,$
$-0.6422, -0.6422, -0.6422, -0.6422, -0.6422, -0.6422, -0.6422,$
$-0.6430, -0.6140)$, $(n = 30)$.

(e) $x_C^* = (-0.3774, -0.5258, -0.4584, -0.6089, -0.5223, -0.6715, -0.5481,$
$-0.6702, -0.5436, -0.6682, -0.5415, -0.6681, -0.5420, -0.6682,$
$-0.5422, -0.6682, -0.5422, -0.6682, -0.5422, -0.6682, -0.5422,$
$-0.6684, -0.5422, -0.6687, -0.5422, -0.6649, -0.5422, -0.6610,$
$-0.5430, -0.6264)$, $(n = 30)$.

13. Another generalization of the Broyden Banded function (BROYDEN2B)

The same as 12 except that $p = 2$ and

(e) $x_C^* = (-0.3774, -0.5209, -0.4584, -0.6014, -0.5223, -0.6643, -0.5481,$
$-0.6630, -0.5436, -0.6611, -0.5415, -0.6611, -0.5420, -0.6612,$
$-0.5422, -0.6612, -0.5422, -0.6612, -0.5422, -0.6612, -0.5422,$
$-0.6613, -0.5422, -0.6616, -0.5422, -0.6585, -0.5422, -0.6557,$
$-0.5430, -0.6228)$, $(n = 30)$.

14. Toint's 7-diagonal generalization of the Broyden Tridiagonal function (TOINTBROY) (see Toint, 1978)

(a) $f(x) = 1 + \sum_{i=1}^{n} |(3 - 2x_i)x_i - x_{i-1} - x_{i+1} + 1|^p + \sum_{i=1}^{n/2} |x_i + x_{i+n/2}|^p$,

where n is even, $p = 7/3$ and $x_0 = x_{n+1} = 0$.

(c) $x^{(0)} = (-1, -1, -1, \ldots, -1)$.

(d) $x_U^* = (-0.4114, -0.4729, -0.4732, -0.4673, -0.4633, -0.4614, -0.4608,$
$-0.4614, -0.4630, -0.4657, -0.4700, -0.4761, -0.4838, -0.4914,$
$-0.4939, -0.4808, -0.4681, -0.4607, -0.4574, -0.4560, -0.4554,$
$-0.4546, -0.4532, -0.4506, -0.4459, -0.4374, -0.4221, -0.3938,$
$-0.3405, -0.2340)$, $(n = 30)$.

(e) $x_C^* = (-0.3114, -0.3802, -0.3732, -0.3780, -0.3632, -0.3712, -0.3608,$
$-0.3713, -0.3630, -0.3758, -0.3700, -0.3867, -0.3838, -0.4029,$
$-0.3939, -0.3919, -0.3681, -0.3706, -0.3574, -0.3658, -0.3554,$
$-0.3643, -0.3532, -0.3601, -0.3459, -0.3469, -0.3221, -0.2973,$
$-0.2405, -0.1811)$, $(n = 30)$.

15. A trigonometric function (TRIG) (Nazareth, 1986)

(a) $f(x) = \sum_{i=1}^{n} [n + i - \sum_{j=1}^{n} (a_{ij} \sin x_j + b_{ij} \cos x_j)]^2$,

where $a_{ij} = \delta_{ij}$, $b_{ij} = 1 + i\delta_{ij}$ and $\delta_{ij} = 1$ if $i = j$ and 0 otherwise.

(c) $x^{(0)} = (1/n, 1/n, 1/n, \ldots, 1/n)$.

(d) $x_U^* = (1.5708, 0.1, 0, 1.5708, 0.1, 0, 1.5708, 0.1, 0, 1.5708)$.

(e) $x_C^* = (1.6708, 0.1, 0.1, 1.5708, 0.2, 0, 1.6708, 0.1, 0.1, 1.5708)$.

16. Another trigonometric function (TOINTTRIG) (Toint, 1978)

(a) $f(x) = \sum_{(i,j) \in J} \alpha_{ij} \sin[\beta_i x_i + \beta_j x_j + \gamma_{ij}]$,

where $\alpha_{ij} = 5[1 + \text{mod}(i, 5) + \text{mod}(j, 5)]$, $\beta_i = 1 + i / 10$ and $\gamma_{ij} = (i + j) / 10$. We selected $J = \{(i, j)$: mod$(|i - j|, 4) = 0\}$.

(c) $x^{(0)} = (1, 1, 1, \ldots, 1)$.

(d) $x_U^* = (2.0511, 1.7968, 1.5817, 1.3973, 1.2375, 1.0976, 0.9742, 0.8645,$
$0.7664, 0.6781)$ $(n = 10)$.

(e) $x_C^* = (2.1511, 1.7968, 1.6817, 1.3973, 1.3375, 1.0976, 1.0742, 0.8645,$
$0.8664, 0.6781)$ $(n = 10)$.

17. A generalization of the Cragg and Levy function (CRAGGLEVY) (Cragg and Levy, 1969)

(a) $f(x) = \sum_{i \in J} [(e^{x_i} - x_{i+1})^4 + 100(x_{i+1} - x_{i+2})^6$
$+ \tan^4(x_{i+2} - x_{i+3}) + x_i^8 + (x_{i+3} - 1)^2]$,

where n is a multiple of 4 and $J = \{1, 5, 9, \ldots, n-3\}$.

(c) $x^{(0)} = (1, 2, 2, 2, 2, 2, 2, 2, \ldots, 2, 2, 2, 2)$,

(d) $x_U^* = (0, 1, 1, 1, \ldots, 0, 1, 1, 1)$,

(e) $x_C^* = (0.1, 1.1045, 1.1, 1.0019, \ldots, 0.1, 1.1045, 1.1, 1.0019)$.

18. A penalty function (PENALTY)

(a) $f(x) = 1 + \sum_{i=1}^{n} x_i + \mu(1 - \sum_{i=1}^{n} 1/x_i)^2 + \mu(1 - \sum_{i=1}^{n} i/x_i)^2$.

For our tests, we selected $\mu = 1000$.

(b) $-0.01 \leq x_i \leq 10000$ for $1 \leq i \leq n$.

(c) $x^{(0)} = (1, 1, 1, \ldots, 1)$,

(d) $x_U^* = 100(0.0371, 0.3346, 0.4718, 0.5772, 0.6662, 0.7446, 0.8155, 0.8807,$
$0.9414, 0.9984, 1.0524, 1.1037, 1.1527, 1.1997, 1.2450)$ $(n = 15)$.

(e) $x_C^* = 100(0.0381, 0.3302, 0.4728, 0.5732, 0.6672, 0.7404, 0.8165, 0.8762,$
$0.9424, 0.9936, 1.0534, 1.0986, 1.1537, 1.1944, 1.2460)$ $(n = 15)$.

19. An Augmented Lagrangian function for a generalization of Hock and Schittkowski's 80–th problem (AUGMLAGN) (Hock and Schittkowski, 1981)

(a) $f(x) = 1 + \sum_{i \in J}[e^{x_i x_{i+1} x_{i+2} x_{i+3} x_{i+4}} + \frac{1}{2}\rho((\sum_{j=0}^{4} x_{i+j}^2 - 10 - \lambda_1)^2 +$
$(x_{i+1}x_{i+2} - 5x_{i+3}x_{i+4} - \lambda_2)^2 + (x_i^3 + x_{i+1}^3 + 1 - \lambda_3)^2)]$,

where n is a multiple of 5 and $J = \{1, 6, 11, \ldots, n-4\}$. For our tests, we selected $\rho = 20$, $\lambda_1 = -0.002008$, $\lambda_2 = -0.001900$ and $\lambda_3 = -0.000261$ (which make $f(x)$ an exact penalty function for Hock and Schittkowski's problem).

(b) $-2.3 \leq x_i \leq 2.3$ for $1 \leq i \leq n$.

(c) $x^{(0)} = (-2, 2, 2, ,-1, -1, -1, -1, 2, -1, -1, \ldots ,-1, -1, 2, -1, -1)$

(d) $x_U^* = (-1.7171, 1.5957, -1.8273, -0.7636, -0.7636, \ldots ,$
$-1.7171, 1.5957, -1.8273, -0.7636, -0.7636)$

(e) $x_C^* = (-1.6171, 1.4782, -1.9928, -0.8877, -0.6636, -1.8045, 1.6957,$
$-1.6488, -0.6636, -0.8425, -0.7290, -0.8553, 2.6161,$
$-1.3343, 0.3363)$ $(n = 15)$.

20. A generalization of a function due to A. Brown (BROWN1) (L. C. W. Dixon, private communication)

(a) $f(x) = [\sum_{i \in J}(x_i - 3)]^2 + \sum_{i \in J}[(0.0001(x_i - 3)^2 - (x_i - x_{i+1}) + e^{20(x_i - x_{i+1})}]$,

where n is a multiple of 2 and $J = \{1, 3, 5, \ldots, n-1\}$.

(b) $-1 \leq x_i \leq 4$ for $1 \leq i \leq n$.

(c) $x^{(0)} = (0, -1, 0, -1, \ldots, 0, -1)$,

(d) $x_U^* = (3, 3.1498, 3, 3.1498, \ldots, 3, 3.1498)$,

(e) $x_C^* = (3.1, 3.2498, 3.1, 3.2498, \ldots, 3.1, 3.2498)$.

21. A generalization of another function due to A. Brown (BROWN3) (L. C. W. Dixon, private communication)

(a) $f(x) = \sum_{i=1}^{n-1}[(x_i^2)^{(x_{i+1}^2 + 1)} + (x_{i+1}^2)^{(x_i^2 + 1)}]$.

(c) $x^{(0)} = (-1, 1, -1, 1, \ldots, -1, 1)$,

(d) $x_U^* = (0, 0, 0, \ldots, 0)$,

(e) $x_C^* = (0.1, 0, 0.1, 0, \ldots, 0.1, 0)$.

22. The discrete boundary value problem (BVP) (Moré, Garbow and Hillstrom, 1981)

(a) $f(x) = \sum_{i=1}^{n}[2x_i - x_{i-1} - x_{i+1} + h^2(x_i + ih + 1)^3 / 2]$,

where $h = 1/(n+1)$ and $x_0 = x_{n+1} = 0$.

(b) $-0.2n \leq x_i \leq 0.2n$ for $1 \leq i \leq n$.

(c) $x_i^{(0)} = i\,h(i\,h - 1)$ for $1 \leq i \leq n$.

(d) $x_U^* = 0.1(-0.4317, -0.8158, -1.1449, -1.4097, -1.5991, -1.6988, -1.6909, -1.5525, -1.2536, -0.7542)$ $(n = 10)$.

$x_U^* = 0.1(0.2321, -0.4520, -0.6588, -0.8514, -1.0288, -1.1895, -1.3322, -1.4553, -1.5571, -1.6354, -1.6881, -1.7127, -1.7060, -1.6650, -1.5856, -1.4636, -1.2938, -1.0702, -0.7858, -0.4323)$ $(n = 20)$.

(e) $x_C^* = 0.01(5.6835, 8.4100, 8.9057, 7.8272, 5.7611, 3.2315, 0.7129, -1.3527, -2.5356, -2.3936)$ $(n = 10)$.

$x_C^* = 0.1(0.7679, 1.3625, 1.8041, 2.1121, 2.3047, 2.3990, 2.4107, 2.3542, 2.2425, 2.0875, 1.9000, 1.6895, 1.4648, 1.2337, 1.0034, 0.7805, 0.5710, 0.3050, 0.2142, 0.0773)$ $(n = 20)$.

23. The discretization of a variational problem (VAR) (Toint, 1978)

(a) $f(x) = 2\sum_{i=1}^{n}[x_i(x_i - x_{i+1})]\,/\,h + 2\lambda h \sum_{i=0}^{n}[(e^{x_{i+1}} - e^{x_i})/(x_{i+1} - x_i)]$,

where $h = 1/(n+1)$ and $x_0 = x_{n+1} = 0$. We selected $\lambda = -3.4$.

(b) $-0.2n \leq x_i \leq 0.2n$ for $1 \leq i \leq n$.

(c) $x_i^{(0)} = 0.1i\,h(1 - i)$ for $1 \leq i \leq n$.

(d) $x_U^* = 0.1(1.4638, 2.8383, 4.1104, 5.2663, 6.2918, 7.1729, 7.8964, 8.4505, 8.8256, 9.0150, 9.0150, 8.8256, 8.4505, 7.8964, 7.1729, 6.2918, 5.2663, 4.1104, 2.8383, 1.4638)$ $(n = 20)$.

$x_U^* = (0.6812, 1.3452, 1.9909, 2.6169, 3.2220, 3.8050, 4.3645, 4.8991, 5.4075, 5.8883, 6.3401, 6.7617, 7.1517, 7.5089, 7.8320, 8.1200, 8.3718, 8.5865, 8.7633, 8.9016, 9.0007, 9.0604, 9.0803, 9.0604, 9.0007, 8.9016, 8.7633, 8.5865, 8.3718, 8.1200, 7.8320, 7.5089, 7.1517, 6.7617, 6.3401, 5.8883, 5.4075, 4.8991, 4.3645, 3.8050, 3.2220, 2.6169, 1.9909, 1.3452, 0.6812)$ $(n = 45)$.

(e) $x_C^* = 0.1(0.2464, 0.4253, 0.5925, 0.7456, 0.8826, 1.0010, 1.0984, 1.1728, 1.2224, 1.2459, 1.2427, 1.2129, 1.1573, 1.0773, 0.9747, 0.8517, 0.7107, 0.5540, 0.3838, 0.1966)$ $(n = 20)$.

$x_C^* = (0.1681, 0.3084, 0.4464, 0.5820, 0.7146, 0.8440, 0.9697, 1.0911, 1.2077, 1.3189, 1.4241, 1.5227, 1.6139, 1.6970, 1.7714, 1.8363, 1.8912, 1.9354, 1.9685, 1.9902, 2.0001, 2.0100, 2.0080, 2.0100, 2.0001, 1.9902, 1.9685, 1.9354, 1.8912, 1.8363, 1.7714, 1.6970, 1.6139, 1.5227, 1.4241, 1.3189, 1.2077, 1.0911, 0.9697, 0.8440, 0.7146, 0.5820, 0.4464, 0.3084, 0.1681)$ $(n = 45)$.

8. References.

D. P. Bertsekas, *Constrained optimization and Lagrange multiplier methods* (1982) Academic Press, London and New York.

A. R. Conn, N. I. M. Gould and Ph. L. Toint, "Global convergence of a class of trust region algorithms for optimization with simple bounds", *Technical Report 86/1, Dept. of*

Maths, FUN, Namur (1986) (Also issued as report CS 86–31, University of Waterloo and report CSS 202, A.E.R.E. Harwell.)

E. E. Cragg and A. V. Levy, "Study on a supermemory gradient method for the minimization of functions", *Journal of Optimization Theory and Applications* **4** (1969) 191–205.

R. S. Dembo, S. C. Eisenstat and T. Steihaug, "Inexact Newton methods", *SIAM Journal on Numerical Analysis* **19** (1982) 400–408.

R. S. Dembo and U. Tulowitzki, "On the minimization of quadratic functions subject to box constraints", Working paper series B 71, (1983) School of Organization and Management, Yale University.

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

R. Fletcher, *Practical Methods of Optimization, Volume 1,* (1980) John Wiley and Sons, Chichester and New York.

R. Fletcher and M. P. Jackson, "Minimization of a quadratic function of many variables subject only to lower and upper bounds", *Journal of the Institute for Mathematics and its Applications* **14** (1974) 159–174.

P. E. Gill and W. Murray, "Minimization subject to bounds on the variables", *Report NAC 71, (1976) National Physical Laboratory, England.*

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

A. Griewank and Ph. L. Toint, "On the unconstrained optimization of partially separable functions", in *Nonlinear Optimization, 1981,* (M. J. D. Powell, ed.) (1982a) 301–312, Academic Press, London and New York.

A. Griewank and Ph. L. Toint, "Partitioned variable metric updates for large structured optimization problems", *Numerische Mathematik* **39** (1982b) 119–137.

W. Hock and K. Schittkowski, *Test Examples for Nonlinear Programming Codes,* Lecture Notes in Economics and Mathematical Systems 187 (1981) Springer–Verlag, Berlin.

J. J. Moré, "Recent developments in algorithms and software for trust region methods", in *Mathematical Programming, the state of the art, Bonn 1982* (A. Bachem, M. Grötschel and B. Korte, eds.) (1983) 258–287, Springer–Verlag, Berlin.

J. J. Moré, B. S. Garbow and K. E. Hillstrom, "Testing unconstrained optimization software", *ACM Transactions on Mathematical Software* **7** (1981) 17–41.

J. L. Nazareth, "Analogues of Dixon's and Powell's theorems for unconstrained minimization with inexact line searches", *SIAM Journal on Numerical Analysis* **23** (1986) 170–177.

D. P. O'leary, "A generalized conjugate gradient algorithm for solving a class of quadratic programming problems", *Linear Algebra and its Applications* **34** (1980) 371–399.

M. J. D. Powell, "A new algorithm for unconstrained optimization", in *Nonlinear*

Programming, (J. B. Rosen, O. L. Mangasarian and K. Ritter, eds.) (1970) 31–65 Academic Press, London and New York.

D. F. Shanno and K. H. Phua, "Matrix conditioning and nonlinear optimization", *Mathematical Programming* **14** (1978) 145–160.

D. C. Sorensen, "An example concerning quasi–Newton estimation of a sparse Hessian", *SIGNUM Newsletter* **16** (1981) 8–10.

T. Steihaug, "The conjugate gradient method and trust regions in large scale optimization", *SIAM Journal on Numerical Analysis* **20** (1983) 626–637.

Ph. L.Toint, "Some numerical results using a sparse matrix updating formula in unconstrained optimization", *Mathematics of Computation* **32** (1978) 839–851.

Ph. L.Toint, "Towards an efficient sparsity exploiting Newton method for minimization", in *Sparse matrices and their uses* (I. S. Duff, ed.) (1981) Academic Press, London and New York.