DAMTP 2003/NA03

On the use of quadratic models in unconstrained minimization without derivatives[1]

M.J.D. Powell

Abstract: Quadratic approximations to the objective function provide a way of estimating first and second derivatives in iterative algorithms for unconstrained minimization. Therefore we address the construction of suitable quadratic models Q by interpolating values of the objective function F. On a typical iteration, the objective function is calculated at the point that minimizes the current quadratic model subject to a trust region bound, and we find that these values of F provide good information for the updating of Q, except that a few extra values are needed occasionally to avoid degeneracy. The number of interpolation points and their positions can be controlled adequately by deleting one of the current points to make room for each new one. An algorithm is described that works in this way. It is applied to an optimization calculation that has between 10 and 160 variables. The numerical results suggest that, if $m=2n+1$, then the number of evaluations of F is only of magnitude n, where m and n are the number of interpolation conditions of each model and the number of variables, respectively. This success is due to the technique that updates Q. It minimizes the Frobenius norm of the change to $\nabla^2 Q$, subject to the interpolation conditions that have been mentioned.

Department of Applied Mathematics and Theoretical Physics,
Centre for Mathematical Sciences,
Wilberforce Road,
Cambridge CB3 0WA,
England.

March, 2003 (Revised September, 2003).

[1]Presented at The First International Conference on Optimization Methods and Software (December, 2002), Hangzhou, China.

1. Least Frobenius norm updating of quadratic models

We consider the efficiency and use of quadratic models in iterative algorithms for unconstrained minimization calculations. Let $F(\underline{x})$, $\underline{x}\in\mathcal{R}^n$, be the objective function whose minimum value is required, let k be the iteration number, and let $\underline{x}_k^*$ be the vector of variables such that $F(\underline{x}_k^*)$ is the least calculated value of the objective function so far at the beginning of the k-th iteration. Then the quadratic model of the k-th iteration is a quadratic polynomial $Q_k(\underline{x})$, $\underline{x}\in\mathcal{R}^n$, that satisfies $Q_k(\underline{x}_k^*)=F(\underline{x}_k^*)$. Each quadratic model is formed automatically. We will give particular attention to the construction of Q_{k+1} from Q_k, which is done by the k-th iteration.

Usually the iteration generates a change to the variables that is based on the approximation

$$Q_k(\underline{x}) \approx F(\underline{x}), \qquad \underline{x}\in\mathcal{R}^n. \tag{1.1}$$

Further, in trust region algorithms it is assumed that this approximation is helpful only if $\underline{x}$ is sufficiently close to $\underline{x}_k^*$. Specifically, a positive parameter Δ_k, called the "trust region radius", is also available at the beginning of the iteration, and, until termination or some special action is required, the next trial vector of variables is $\underline{x}_k^*+\underline{d}_k$, where $\underline{d}_k$ is an estimate of the vector $\underline{d}$ that solves the subproblem

$$\text{Minimize} \quad Q_k(\underline{x}_k^*+\underline{d}) \quad \text{subject to} \quad \|\underline{d}\| \le \Delta_k, \tag{1.2}$$

the vector norm being Euclidean. There are good ways of generating $\underline{d}_k$ that do not require the second derivative matrix $\nabla^2 Q_k$ to be positive definite (see Moré and Sorensen, 1983, for instance). Then the new function value $F(\underline{x}_k^*+\underline{d}_k)$ may be calculated, and $\underline{x}_{k+1}^*$ may be defined by the formula

$$\underline{x}_{k+1}^* = \begin{cases} \underline{x}_k^*, & F(\underline{x}_k^*+\underline{d}_k)\ge F(\underline{x}_k^*) \\ \underline{x}_k^*+\underline{d}_k, & F(\underline{x}_k^*+\underline{d}_k)< F(\underline{x}_k^*). \end{cases} \tag{1.3}$$

Equation (1.3) provides the best vector of variables so far, but many algorithms set $\underline{x}_{k+1}^*$ to $\underline{x}_k^*+\underline{d}_k$ only if the reduction $F(\underline{x}_k^*)-F(\underline{x}_k^*+\underline{d}_k)$ is sufficiently large. Originally this device was introduced to assist proofs of convergence, but now the use of sufficient reductions has become standard practice. In Sections 2 and 3, however, we are going to study a trust region algorithm that retains formula (1.3), because we welcome every decrease that occurs in the value of the objective function. The author does not know of cases where formula (1.3) prevents convergence, provided that the new trust region radius Δ_{k+1} is chosen carefully. For example, the value $\Delta_{k+1}=\frac{1}{2}\|\underline{d}_k\|$ may be suitable if the condition

$$F(\underline{x}_k^*) - F(\underline{x}_k^*+\underline{d}_k) < 0.1\,[\,Q_k(\underline{x}_k^*) - Q_k(\underline{x}_k^*+\underline{d}_k)\,] \tag{1.4}$$

holds, because then the iteration fails to achieve one tenth of the reduction in the objective function that is predicted by the quadratic model. On the other hand, if the change in F compares favourably with the predicted change, then the choice $\Delta_{k+1} = \max[\Delta_k, 2\|\underline{d}_k\|]$ may be made. Details of this kind are addressed in many publications, including the book of Conn, Gould and Toint (2000).

Several methods have been proposed for constructing the new quadratic model Q_{k+1} when first derivatives of the objective function are available (see Fletcher, 1987, for instance). They satisfy $Q_{k+1}(\underline{x}^*_{k+1}) = F(\underline{x}^*_{k+1})$ and $\underline{\nabla} Q_{k+1}(\underline{x}^*_{k+1}) = \underline{\nabla} F(\underline{x}^*_{k+1})$, so Q_{k+1} has the form

$$Q_{k+1}(\underline{x}^*_{k+1}+\underline{d}) \;=\; F(\underline{x}^*_{k+1}) + \underline{d}^T \underline{\nabla} F(\underline{x}^*_{k+1}) + \tfrac{1}{2}\, \underline{d}^T B_{k+1}\, \underline{d}, \qquad \underline{d} \in \mathcal{R}^n, \tag{1.5}$$

for some $n \times n$ symmetric matrix B_{k+1}. The usual choices of B_{k+1} depend on the remark that, if F is twice differentiable, then its second derivatives have the property

$$\Big\{ \int_0^1 \nabla^2 F(\underline{x}^*_k + \theta\, \underline{d}_k)\, d\theta \Big\}\, \underline{d}_k \;=\; \underline{\nabla} F(\underline{x}^*_k + \underline{d}_k) - \underline{\nabla} F(\underline{x}^*_k). \tag{1.6}$$

We see that the left hand side is the product $\nabla^2 F \underline{d}_k$ if F happens to be a quadratic polynomial, where $\nabla^2 F$ is the constant second derivative matrix of F. Therefore the difference in gradients on the right hand side is calculated, and B_{k+1} is required to satisfy the condition

$$B_{k+1}\, \underline{d}_k \;=\; \underline{\nabla} F(\underline{x}^*_k + \underline{d}_k) - \underline{\nabla} F(\underline{x}^*_k). \tag{1.7}$$

A good discussion of the merits of this technique can be found in Dennis and Schnabel (1983).

Equation (1.7) provides only n constraints on the new model (1.5). Because B_{k+1} is an $n \times n$ symmetric matrix, the number of remaining degrees of freedom is $\frac{1}{2} n(n-1)$, which are fixed by different methods in different ways. In particular, the well-known symmetric Broyden formula is relevant to our work. It takes up the freedom by minimizing the Frobenius norm

$$\| B_{k+1} - B_k \|_F \;=\; \| \nabla^2 Q_{k+1} - \nabla^2 Q_k \|_F \tag{1.8}$$

of the difference between the second derivative matrices of the old and new quadratic models. The Frobenius norm of a matrix is the square root of the sum of squares of its elements.

Thus the symmetric Broyden formula enjoys the highly useful property that is given in the theorem below. The theorem is presented in a way that is also applicable to the method for unconstrained minimization without derivatives that is going to be studied in Sections 2 and 3. Specifically, we write Q_{k+1} in the form

$$Q_{k+1}(\underline{x}) \;=\; \sum_{j=1}^{m^*} \mu_j\, b_j(\underline{x}), \qquad \underline{x} \in \mathcal{R}^n, \tag{1.9}$$

where b_j, $j = 1, 2, \ldots, m^*$, is a basis of the linear space of polynomials of degree at most two from $\mathcal{R}^n$ to $\mathcal{R}$, the value of m^* being $\frac{1}{2}(n+1)(n+2)$. Then the condition

(1.7) on $B_{k+1} = \nabla^2 Q_{k+1}$ provides n linear equality constraints on the coefficients μ_j, $j=1,2,\ldots,m^*$. Further, the properties

$$Q_{k+1}(\underline{x}^*_{k+1}) \;=\; F(\underline{x}^*_{k+1}) \quad \text{and} \quad \underline{\nabla} Q_{k+1}(\underline{x}^*_{k+1}) \;=\; \underline{\nabla} F(\underline{x}^*_{k+1}) \tag{1.10}$$

are also linear equality constraints on the coefficients of expression (1.9). We let $\mathcal{S}$ be the set of polynomials of the form (1.9) that satisfy these constraints, so $\mathcal{S}$ is a linear manifold. It follows from equations (1.6) and (1.10) that, if F happens to be a quadratic polynomial, then F is also an element of $\mathcal{S}$.

Theorem: Let $F(\underline{x})$, $\underline{x} \in \mathcal{R}^n$, be a quadratic polynomial, and let the old quadratic model Q_k be available. The new model Q_{k+1} has to satisfy the linear constraints $Q_{k+1} \in \mathcal{S}$, as stated in the previous paragraph, and the remaining freedom in Q_{k+1} has to make the Frobenius norm $\|\nabla^2 Q_{k+1} - \nabla^2 Q_k\|_F$, $Q_{k+1} \in \mathcal{S}$, as small as possible. Then, if F is in $\mathcal{S}$, this construction provides the relation

$$\|\nabla^2 Q_{k+1} - \nabla^2 F\|_F^2 \;=\; \|\nabla^2 Q_k - \nabla^2 F\|_F^2 \;-\; \|\nabla^2 Q_{k+1} - \nabla^2 Q_k\|_F^2. \tag{1.11}$$

Proof: Equation (1.11) is a well-known consequence of orthogonal projection within an inner product space. Specifically, we employ the inner product that is induced by the semi-norm $\|Q\| = \|\nabla^2 Q\|_F$, $Q \in \mathcal{Q}$, where $\mathcal{Q}$ is the linear space of polynomials of degree at most two. Thus Q_{k+1} is the orthogonal projection of Q_k onto the linear manifold $\mathcal{S} \subset \mathcal{Q}$, which gives the required result. $\square$

The theorem implies the inequality

$$\|\nabla^2 Q_{k+1} - \nabla^2 F\|_F \;\leq\; \|\nabla^2 Q_k - \nabla^2 F\|_F, \tag{1.12}$$

when F is a quadratic function. Thus, if the symmetric Broyden formula is employed on every iteration, then the Frobenius norm of the error of the approximation $\nabla^2 Q_k \approx \nabla^2 F$ decreases monotonically during the iterative procedure. This property is welcome, but at first sight does not cause much enthusiasm, because the assumption that F is quadratic seems to be very restrictive, and because $\|\nabla^2 Q_k - \nabla^2 F\|_F$ may stay bounded away from zero as $k \to \infty$. On the other hand, the theorem also provides the limit

$$\lim_{k\to\infty} \|\nabla^2 Q_{k+1} - \nabla^2 Q_k\|_F \;=\; 0, \tag{1.13}$$

which causes $\underline{x}_k$, $k=1,2,3,\ldots$, to converge to the optimal vector of variables at a superlinear rate, when the objective function is quadratic, for many of the usual choices of the changes $\underline{d}_k$ to the variables.

These changes are derived from the quadratic models, and then the standard way of proving superlinear convergence, due to Broyden, Dennis and Moré (1973), requires the condition

$$\lim_{k\to\infty} \|\underline{\nabla} F(\underline{x}^*_k + \underline{d}_k) - \underline{\nabla} Q_k(\underline{x}^*_k + \underline{d}_k)\| \,/\, \|\underline{d}_k\| \;=\; 0. \tag{1.14}$$

In other words, as k becomes large, the approximation (1.1) has to provide a good prediction of the new gradient $\underline{\nabla} F(\underline{x}_k^*+\underline{d}_k)$. Now the property $\underline{\nabla} Q_k(\underline{x}_k^*)=\underline{\nabla} F(\underline{x}_k^*)$, the constraint (1.7) on $B_{k+1}=\nabla^2 Q_{k+1}$, and $Q_k \in \mathcal{Q}$ imply the identity

$$\begin{aligned} &\|\underline{\nabla} F(\underline{x}_k^*+\underline{d}_k) - \underline{\nabla} Q_k(\underline{x}_k^*+\underline{d}_k)\| \,/\, \|\underline{d}_k\| \\ &= \|\{\underline{\nabla} F(\underline{x}_k^*+\underline{d}_k) - \underline{\nabla} F(\underline{x}_k^*)\} - \{\underline{\nabla} Q_k(\underline{x}_k^*+\underline{d}_k) - \underline{\nabla} Q_k(\underline{x}_k^*)\}\| \,/\, \|\underline{d}_k\| \\ &= \|(\nabla^2 Q_{k+1} - \nabla^2 Q_k)\, \underline{d}_k\| \,/\, \|\underline{d}_k\|. \end{aligned} \tag{1.15}$$

It follows from equation (1.13) that the limit (1.14) is achieved. This argument is also valid for many other objective functions that have continuous second derivatives, because then equation (1.11) holds if $\nabla^2 F$ is the matrix inside the braces of expression (1.6), but careful attention has to be given to the changes to $\nabla^2 F$ that occur in expression (1.11) during the sequence of iterations. The purpose of these remarks is to point out that the least Frobenius norm method for updating quadratic models may provide fast convergence, even if $\|\nabla^2 Q_k - \nabla^2 F(\underline{x}_k^*)\|_F$ does not become small as k is increased.

Throughout the remainder of this paper, we study trust region methods when $F(\underline{x})$, $\underline{x}\in\mathcal{R}^n$, is a general function, and its derivatives are not available. Therefore the quadratic models are derived from interpolation equations. We let the conditions on Q_{k+1} be the constraints

$$Q_{k+1}(\underline{x}_i^+) \;=\; F(\underline{x}_i^+), \qquad i=1,2,\ldots,m, \tag{1.16}$$

where m is a prescribed integer that is independent of k, but the positions of the interpolation points $\underline{x}_i^+ \in \mathcal{R}^n$, $i=1,2,\ldots,m$, are generated automatically. They include the point $\underline{x}_{k+1}^*$, in order that $Q_{k+1}(\underline{x}_{k+1}^*)=F(\underline{x}_{k+1}^*)$ holds, and also they must have the following two properties:

(1) The equations (1.16) can be satisfied by a model $Q_{k+1}\in\mathcal{Q}$ for any right hand sides.

(2) The interpolation points do not all lie in a hyperplane $\{\underline{x} : \pi(\underline{x})=0\}$, where π is a linear polynomial.

It follows that, given the old model Q_k, there is a unique new model Q_{k+1} that minimizes $\|\nabla^2 Q_{k+1} - \nabla^2 Q_k\|_F$, $Q_{k+1}\in\mathcal{Q}$, subject to the constraints (1.16). We address algorithms for unconstrained minimization without derivatives that update quadratic models in this way.

Condition **(2)** of the last paragraph implies $m\geq n+1$. It also implies that, if $m=n+1$ is chosen, then there is enough freedom in the constant and linear terms of Q_{k+1} to satisfy the equations (1.16). Thus the least Frobenius norm updating method yields $\nabla^2 Q_{k+1} = \nabla^2 Q_k$, so equation (1.11) becomes trivial. Moreover, condition **(1)** implies $m\leq m^*=\frac{1}{2}(n+1)(n+2)$. Further, if $m=m^*$ is chosen, then all of the freedom in the new model is taken up by the interpolation conditions. Hence Q_{k+1} is independent of Q_k, and $\nabla^2 Q_{k+1} = \nabla^2 F$ occurs if F is quadratic, so again equation (1.11) becomes trivial. We assume that m is picked from the interval $[n{+}2, m^*]$, in order that some changes are made to the second derivatives of the current quadratic model during the minimization calculation.

We now let $\mathcal{S}$ be the linear manifold of functions of the form (1.9) that satisfy the linear equality constraints (1.16). It follows from the theorem of this section that, when F is a quadratic function, the updating method provides the limit (1.13). Therefore, in view of the remarks above on convergence when first derivatives of F are available, we are optimistic that a version of the method without derivatives can be developed that enjoys excellent convergence properties in practice. Theoretical analysis is more difficult than before, however, because $\nabla^2 Q_{k+1} = \nabla^2 Q_k$ is possible when a change to the second derivatives is expected, an example being given below. Therefore we are going to draw our conclusions from the numerical experiments of Section 3, using the algorithm that is the subject of Section 2.

The algorithm retains only m values of F for the beginning of the k-th iteration, namely the right hand sides of the interpolation equations

$$Q_k(\underline{x}_i) \;=\; F(\underline{x}_i), \qquad i=1,2,\ldots,m, \tag{1.17}$$

that are satisfied by Q_k. Further, except on a few iterations that do not change the quadratic model, the k-th iteration calculates just one new function value, namely $F(\underline{x}_k^* + \underline{d}_k)$, where $\underline{x}_k^* + \underline{d}_k$ is not in the set $\{\underline{x}_i : i=1,2,\ldots,m\}$. Therefore at most one interpolation is moved, say $\underline{x}_t$, and then the new points have the positions

$$\underline{x}_i^+ = \underline{x}_i, \quad i \in \{1,2,\ldots,m\} \backslash \{t\} \qquad \text{and} \qquad \underline{x}_t^+ = \underline{x}_k^* + \underline{d}_k, \tag{1.18}$$

for some integer t in $[1, m]$. It follows that the updating technique provides the new model

$$Q_{k+1}(\underline{x}) \;=\; Q_k(\underline{x}) + \{F(\underline{x}_t^+) - Q_k(\underline{x}_t^+)\}\, \ell_t^+(\underline{x}), \qquad \underline{x} \in \mathcal{R}^n, \tag{1.19}$$

where ℓ_t^+ is the element of $\mathcal{Q}$ such that $\|\nabla^2 \ell_t^+\|_F$ is least subject to the Lagrange conditions

$$\ell_t^+(\underline{x}_i^+) \;=\; \delta_{it}, \qquad i=1,2,\ldots,m, \tag{1.20}$$

where δ_{it} is the “Kronecker delta”.

We end this section with the example that demonstrates $\nabla^2 Q_{k+1} = \nabla^2 Q_k$. We let n and m have the values 2 and 4, respectively, and we let the points $\underline{x}_i$, $i=1,2,3$, lie on the line $\{\underline{x} : \pi(\underline{x})=0\}$, where π is a linear polynomial, but $\pi(\underline{x}_4)$ is nonzero, due to condition **(2)** on the old interpolation points. We also assume $\pi(\underline{x}_k^* + \underline{d}_k) \neq 0$, which permits $t=4$ in formula (1.18). This choice would give

$\nabla^2 Q_{k+1} = \nabla^2 Q_k$, however, because the function ℓ_t^+ in expression (1.19) would be the linear polynomial

$$\ell_t^+(\underline{x}) \;=\; \pi(\underline{x}) \,/\, \pi(\underline{x}_t^+), \qquad \underline{x} \in \mathcal{R}^2. \tag{1.21}$$

On the other hand, each of the choices $t=1$, $t=2$ and $t=3$ is also allowed by conditions **(1)** and **(2)** on the new interpolation points, and in these cases $F(\underline{x}_k^* + \underline{d}_k) \neq Q_k(\underline{x}_k^* + \underline{d}_k)$ would imply $\nabla^2 Q_{k+1} \neq \nabla^2 Q_k$. The point of the example is that the algorithm of Section 2 is designed to be robust when the objective function is noisy. Therefore it includes a feature that keeps the interpolation points well separated automatically. Thus $t=4$ is mandatory in the example if $\underline{x}_k^* + \underline{d}_k$ is sufficiently close to $\underline{x}_4$. Then the replacement of one of the collinear interpolation points has to wait until a later iteration, which postpones any improvement to the second derivative matrix of the quadratic model.

2. An algorithm for unconstrained minimization without derivatives

Recently the author developed the algorithm that has the name UOBYQA, which is an acronym for Unconstrained Optimization BY Quadratic Approximation. It has the form that is described in Section 1, except that the least Frobenius norm updating method is not employed, because on each iteration the new quadratic model Q_{k+1} is defined uniquely by the interpolation conditions (1.16), the value of m being $m = m^* = \frac{1}{2}(n+1)(n+2)$. It follows that the routine work of each iteration is of magnitude n^4, which is tolerable for $n \leq 25$, but the total amount of computation becomes prohibitive for large n. Therefore the author investigated the possibility of forming useful quadratic models from only $\mathcal{O}(n)$ function values. That task is straightforward if $\nabla^2 F$ is known to have only $\mathcal{O}(n)$ nonzero elements, because most of the freedom in each new model Q_{k+1} can be taken up by including the sparsity pattern of $\nabla^2 F$ in $\nabla^2 Q_{k+1}$, so m is less than m^* by the number of independent sparsity conditions. One can also impose a sparsity pattern on $\nabla^2 Q_{k+1}$ that is not present in $\nabla^2 F$. Indeed, at a conference in Kyoto in 2001, the author presented numerical results for a version of UOBYQA with $m = 2n + 1$, that value being appropriate because the second derivative matrix of every quadratic model is forced to be diagonal, even if none of the elements of $\nabla^2 F$ is zero. Thus the amount of work of each iteration is reduced to only $\mathcal{O}(n^2)$, including the approximate solution of the trust region subproblem (1.2). The algorithm was applied successfully to some unconstrained minimization calculations without derivatives that have more than 100 variables.

In all of those methods, the number of interpolation conditions is the number of degrees of freedom in each new quadratic model, and the construction of Q_{k+1} does not depend on Q_k. Indeed, the least Frobenius norm updating method was not tried by the author until January, 2002. It gave some results that seemed to deserve publicity as soon as possible, so they are included in the author's contribution to the proceedings of the Kyoto conference (Powell, 2003), although the work was done after the meeting. Those activities are mentioned, in order to explain that much software for trust region methods with quadratic models was available at the beginning of the work on least Frobenius norm updating. The algorithm that generated the results of Section 3 is based on UOBYQA (Powell, 2002a), the changes that were made being described below. Many details and some analysis of the new updating method are published elsewhere (Powell, 2002b), but that paper is without a specific algorithm for unconstrained optimization. The new algorithm is still under development, because occasionally the efficiency of the current version is damaged severely by the effects of computer rounding errors.

As in UOBYQA, the user of the current algorithm has to provide an initial vector of variables, $\underline{x}_0^*$ say, the initial and final values of the trust region radius, $\rho_{\rm beg}$ and $\rho_{\rm end}$ say, and a subroutine that calculates $F(\underline{x})$ for any vector of variables $\underline{x}\in\mathcal{R}^n$. Now, however, the number m of interpolation conditions has to be given too. Let $\underline{x}_1$ be $\underline{x}_0^*$ and, for $j=1,2,\ldots,n$, let $\underline{x}_{j+1}$ and $\underline{x}_{j+n+1}$ be $\underline{x}_0^*+\rho_{\rm beg}\underline{e}_j$ and $\underline{x}_0^*-\rho_{\rm beg}\underline{e}_j$, respectively, where $\underline{e}_j$ is the j-th coordinate vector in $\mathcal{R}^n$. If $m\le 2n+1$, the initial interpolation points are the first m points of the sequence $\underline{x}_i$, $i=1,2,\ldots,2n+1$. Otherwise, the sequence is augmented by $m-2n-1$ points of the form $\underline{x}_0^*\pm\rho_{\rm beg}\underline{e}_\alpha\pm\rho_{\rm beg}\underline{e}_\beta$, where α and β are different integers from $[1,n]$, and where the $m-2n-1$ choices of the pairs $\{\alpha,\beta\}$ are all different. The values $F(\underline{x}_i)$, $i=1,2,\ldots,m$, are computed. Then the quadratic model Q_1 of the first iteration is formed by minimizing $\|\nabla^2 Q_1\|_F$, $Q_1\in\mathcal{Q}$, subject to the equations

$$Q_1(\underline{x}_i) \;=\; F(\underline{x}_i), \qquad i=1,2,\ldots,m, \tag{2.1}$$

which requires only $\mathcal{O}(n^2)$ operations, because of the positions of the initial interpolation points. Further, $\underline{x}_1^*$ is set to the first point in the sequence $\underline{x}_i$, $i=1,2,\ldots,m$, that provides the least value of the objective function so far, and $\Delta_1=\rho=\rho_{\rm beg}$ is set too. Moreover, an $(m+n+1)\times(m+n+1)$ symmetric matrix H is constructed, which gives the coefficients of the Lagrange functions for use in formula (1.19) as described later. These operations generate the data for the first iteration of the current algorithm.

The adjustment of the parameters Δ_k and ρ is taken from UOBYQA, Δ_k being required for the trust region subproblem (1.2). The purpose of ρ is that it is a lower bound on Δ_k that is never increased. It is reduced automatically from $\rho_{\rm beg}$ to $\rho_{\rm end}$, each decrease being by about a factor of ten, when the condition $\Delta_k\ge\rho$ seems to be preventing further progress. If $\|\underline{d}_k\|<\frac{1}{2}\rho$ occurs for the current ρ, then, because the difference $F(\underline{x}_k^*+\underline{d}_k)-F(\underline{x}_k^*)$ may be influenced strongly by rounding errors, the value $F(\underline{x}_k^*+\underline{d}_k)$ is not calculated. These devices assist the robustness of the algorithm when the objective function is noisy. Every iteration employs a trust region step until one of the inequalities

$$\|\underline{d}_k\| < \tfrac{1}{2}\,\rho \quad\text{or}\quad F(\underline{x}_k^*+\underline{d}_k) > F(\underline{x}_k^*) - \tfrac{1}{2}\,[\,Q_k(\underline{x}_k^*) - Q_k(\underline{x}_k^*+\underline{d}_k)\,] \tag{2.2}$$

is obtained. Then either the work with the current ρ is complete, or the step $\underline{d}_{k+1}$ of the next iteration may not be a trust region step, being chosen instead to improve the positions of the interpolation points as described below.

When $\|\underline{d}_k\| < \frac{1}{2}\rho$ occurs, then $\nabla^2 Q_k$ is positive definite, and its least eigenvalue, η_k say, is estimated. We take the view that the work with the current ρ is complete if recently observed errors in quadratic models are at most $\frac{1}{2}\eta_k\rho^2$. Therefore we let $\mathcal{J}_k$ be the set of integers j from the interval $[1, k-1]$ such that $F(\underline{x}_j + \underline{d}_j)$ is calculated and such that all of the trust region bounds Δ_i, $j \le i \le k$ have the value ρ. The errors $\varepsilon_j = |F(\underline{x}_j + \underline{d}_j) - Q_j(\underline{x}_j + \underline{d}_j)|$, $j \in \mathcal{J}_k$, are noted. We also let $\underline{x}_i^+$, $i = 1, 2, \ldots, m$, be the interpolation points for the beginning of the $(k+1)$-th iteration, which are chosen in a way that is addressed later. They are the same as $\underline{x}_i$, $i = 1, 2, \ldots, m$, in the case $\|\underline{d}_k\| < \frac{1}{2}\rho$, but, when $F(\underline{x}_k^* + \underline{d}_k)$ is computed, one of the interpolation points of Q_k may be replaced by $\underline{x}_k^* + \underline{d}_k$. Further, $\underline{x}_{k+1}^*$ is in the set $\{\underline{x}_i^+ : i = 1, 2, \ldots, m\}$, and is the oldest interpolation point such that $F(\underline{x}_{k+1}^*)$ is the least calculated value of F so far.

The work of the algorithm with the current ρ is complete if $\|\underline{d}_k\| < \frac{1}{2}\rho$ holds, if $\mathcal{J}_k$ contains at least three integers, and if all the last three values of ε_j, $j \in \mathcal{J}_k$, satisfy $\varepsilon_j \le \frac{1}{2}\eta_k\rho^2$. It is also complete if one of the inequalities (2.2) is obtained, and if the new interpolation points have the property

$$\|\underline{x}_i^+ - \underline{x}_{k+1}^*\| \;\le\; 2\rho, \qquad i = 1, 2, \ldots, m, \tag{2.3}$$

this test being taken from UOBYQA. The other tests on the suitability of the quadratic model are different in UOBYQA, however, because they depend on the remark that its models are exact when F is quadratic. The next step of our algorithm after completing the iterations with the current ρ is the same as in UOBYQA. Indeed, termination occurs if ρ is at its final value ρ_{end}, but otherwise there are more iterations with a smaller ρ.

Our algorithm makes the following choice of $\underline{d}_{k+1}$ when one of the inequalities (2.2) occurs and another iteration is required without changing ρ. If the bounds

$$\|\underline{x}_i^+ - \underline{x}_{k+1}^*\| \;\le\; 2\Delta_{k+1}, \qquad i = 1, 2, \ldots, m, \tag{2.4}$$

hold, Δ_{k+1} being given by the method of UOBYQA, then the sequence of trust region steps is continued, because the new model Q_{k+1} is expected to be adequate for the new trust region radius Δ_{k+1}. Otherwise the $(k+1)$-th iteration will replace $\underline{x}_t^+$ by $\underline{x}_{k+1}^* + \underline{d}_{k+1}$, where $\underline{x}_t^+$ is a point of the set $\{\underline{x}_i^+ : i = 1, 2, \ldots, m\}$ that is furthest from $\underline{x}_{k+1}^*$. In this case the main purpose of $\underline{d}_{k+1}$ is to assist the construction of the quadratic model Q_{k+2} by the $(k+1)$-th iteration. Hence we require the least Frobenius norm updating formula of that iteration to have a large denominator, which is achieved by a large value of $|\ell_t^+(\underline{x}_{k+1}^* + \underline{d}_{k+1})|$ (Powell, 2002b), the Lagrange function ℓ_t^+ being defined in the penultimate paragraph of Section 1. Therefore we let $\underline{d}_{k+1}$ be an estimate of the vector $\underline{d}$ that solves the subproblem

$$\text{Maximize} \quad |\ell_t^+(\underline{x}_{k+1}^*+\underline{d})| \quad \text{subject to} \quad \|\underline{d}\| \le \max[0.1\Delta_{k+1}, \rho]\,. \tag{2.5}$$

Further, if the $(k+1)$-th iteration works in this way, then the $(k+2)$-th iteration begins a new sequence of iterations that derive their trial steps from the trust region subproblem (1.2).

We now turn to the storage and updating of all the coefficients of all the Lagrange functions, because some of them are required by formula (1.19) and the subproblem (2.5). We consider the Lagrange functions ℓ_j, $j=1,2,\ldots,m$, of the interpolation points $\underline{x}_i$, $i=1,2,\ldots,m$, at the beginning of the k-th iteration. Therefore ℓ_j is the unique quadratic polynomial that minimizes $\|\nabla^2\ell_j\|_F$ subject to the constraints

$$\ell_j(\underline{x}_i) \;=\; \delta_{ij}, \qquad 1\le i,j\le m. \tag{2.6}$$

The KKT conditions of this problem provide the form

$$\nabla^2\ell_j \;=\; \sum_{s=1}^m H_{sj}\,(\underline{x}_s-\underline{x}_0)\,(\underline{x}_s-\underline{x}_0)^T, \qquad j=1,2,\ldots,m, \tag{2.7}$$

where $\underline{x}_0$ can be any fixed vector in $\mathcal{R}^n$, because the KKT conditions also give the equations

$$\sum_{s=1}^m H_{sj}=0 \quad \text{and} \quad \sum_{s=1}^m H_{sj}\,\underline{x}_s = \sum_{s=1}^m H_{sj}(\underline{x}_s-\underline{x}_0)=0, \quad j=1,2,\ldots,m, \tag{2.8}$$

(Powell, 2002b). The algorithm picks $\underline{x}_0=\underline{x}_0^*$ initially and adjusts $\underline{x}_0$ occasionally, because smaller values of $\|\underline{x}_s-\underline{x}_0\|$, $s=1,2,\ldots,m$, reduce the contributions from computer rounding errors to expression (2.7).

Having chosen $\underline{x}_0$, we let ℓ_j be the quadratic polynomial

$$\ell_j(\underline{x}) \;=\; c_j + (\underline{x}-\underline{x}_0)^T\underline{g}_j + \tfrac{1}{2}\,(\underline{x}-\underline{x}_0)^T\,\nabla^2\ell_j\,(\underline{x}-\underline{x}_0), \qquad \underline{x}\in\mathcal{R}^n, \tag{2.9}$$

which has $m+n+1$ parameters, namely c_j, the components of $\underline{g}_j$, and the coefficients H_{sj}, $s=1,2,\ldots,m$, of formula (2.7). These parameters are constrained by equations (2.6) and (2.8). Expressions (2.7) and (2.9) allow the Lagrange conditions (2.6) to be written as the identities

$$c_j + (\underline{x}_i-\underline{x}_0)^T\underline{g}_j + \tfrac{1}{2}\sum_{s=1}^m H_{sj}\,\{(\underline{x}_i-\underline{x}_0)^T(\underline{x}_s-\underline{x}_0)\}^2 \;=\; \delta_{ij}, \quad 1\le i,j\le m. \tag{2.10}$$

It follows that the parameters of ℓ_j satisfy the square system of linear equations

$$\left(\begin{array}{c|c} A & \underline{e}\;\;X^T \\ \hline \underline{e}^T & \\ X & 0 \end{array}\right)\left(\begin{array}{c} H_{1j} \\ \vdots \\ H_{mj} \\ \hline c_j \\ \underline{g}_j \end{array}\right) \;=\; W\left(\begin{array}{c} H_{1j} \\ \vdots \\ H_{mj} \\ \hline c_j \\ \underline{g}_j \end{array}\right) \;=\; \underline{e}_j, \tag{2.11}$$

where A is the $m\times m$ matrix with the elements $\frac{1}{2}\{(\underline{x}_i-\underline{x}_0)^T(\underline{x}_s-\underline{x}_0)\}^2$, $1\le i,s\le m$, where $\underline{e}$ is the vector of ones in $\mathcal{R}^m$, where X is the $n\times m$ matrix with the columns $\underline{x}_i-\underline{x}_0$, $i=1,2,\ldots,m$, where W is defined to be the $(m+n+1)\times(m+n+1)$ matrix on the left hand side, and where $\underline{e}_j$ on the right hand side is the j-th coordinate vector in $\mathcal{R}^{m+n+1}$. It can be shown that W, which is symmetric and independent of j, is nonsingular, because of conditions **(1)** and **(2)** of Section 1. Therefore the parameters of ℓ_j are the elements of the j-th column of W^{-1}. The algorithm generates all the elements of W^{-1} explicitly, so from now on we change the definition of H to $H=W^{-1}$, which preserves formula (2.7). We note that H is symmetric. Moreover, the matrix H for the first iteration is constructed in only $\mathcal{O}(m^2)$ operations, by selecting $\underline{x}_0=\underline{x}_0^*$ and by taking advantage of the sparsity in W that is given by the first set of interpolation points, where we are recalling the procedure of the third paragraph of this section.

Another advantage of the availability of H is that, for any $\underline{x}$ in $\mathcal{R}^n$, all the values $\ell_i(\underline{x})$, $i=1,2,\ldots,m$, can be calculated in only $\mathcal{O}(m^2)$ operations. Specifically, we let $\underline{v}=\underline{v}(\underline{x})$ be the vector in $\mathcal{R}^{m+n+1}$ that has the components

$$\left.\begin{array}{ll} v_s=\frac{1}{2}\{(\underline{x}-\underline{x}_0)^T(\underline{x}_s-\underline{x}_0)\}^2, & s=1,2,\ldots,m, \\ v_{m+1}=1 \quad \text{and} \quad v_{m+i+1}=(\underline{x}-\underline{x}_0)_i, & i=1,2,\ldots,n. \end{array}\right\} \tag{2.12}$$

Then, because of equations (2.7) and (2.9), the required numbers $\ell_i(\underline{x})$, $i=1,2,\ldots,m$, are the first m components of the product $H\underline{v}$. For example, if $\underline{x}$ is the interpolation point $\underline{x}_j$, then $\underline{v}=\underline{v}(\underline{x}_j)$ is the j-th column of W, so the construction provides the vector $H\underline{v}=HW\underline{e}_j=\underline{e}_j$, which agrees with the Lagrange conditions $\ell_i(\underline{x}_j)=\delta_{ij}$, $i=1,2,\ldots,m$.

The matrix H is updated as follows when the change (1.18) is made to the interpolation points. Let W^+ and $H^+=(W^+)^{-1}$ be the new versions of W and H. Fortunately, because only the interpolation point $\underline{x}_t$ is moved, the nonzero elements of the matrix W^+-W are confined to its t-th row and column. Hence the rank of W^+-W is at most two, so the calculation of H^+ from H requires only $\mathcal{O}(m^2)$ operations. We put $\underline{x}=\underline{x}_t^+=\underline{x}_k^*+\underline{d}_k$ into expression (2.12), because the resultant vector $\underline{v}=\underline{v}(\underline{x}_k^*+\underline{d}_k)$ is the t-th column of W^+, except for the diagonal element

$$W_{tt}^+ = \tfrac{1}{2}\,\|\underline{x}_t^+-\underline{x}_0\|^4 = \tfrac{1}{2}\,\|\underline{x}_k^*+\underline{d}_k-\underline{x}_0\|^4. \tag{2.13}$$

Further, according to formula (4.14) of Powell (2002b), H^+ is the matrix

$$\begin{aligned} H^+ = H + \frac{1}{\sigma_t}\Big[& \alpha_t\,(\underline{e}_t-H\,\underline{v})\,(\underline{e}_t-H\,\underline{v})^T - \beta_t\,H\,\underline{e}_t\,\underline{e}_t^T H \\ & + \tau_t\Big\{H\,\underline{e}_t\,(\underline{e}_t-H\,\underline{v})^T + (\underline{e}_t-H\,\underline{v})\,\underline{e}_t^T H\Big\}\Big], \end{aligned} \tag{2.14}$$

where the definitions of α_t, β_t, τ_t and σ_t occur in the formulae

$$\left.\begin{array}{ll} \alpha_i=\underline{e}_i^T H\,\underline{e}_i=H_{ii}, & \beta_i=\frac{1}{2}\,\|\underline{x}_k^*+\underline{d}_k-\underline{x}_0\|^4-\underline{v}^T H\,\underline{v} \\ \tau_i=\underline{e}_i^T H\,\underline{v}=\ell_i(\underline{x}_k^*+\underline{d}_k) & \text{and} \quad \sigma_i=\alpha_i\beta_i+\tau_i^2 \end{array}\right\}, \quad i=1,2,\ldots,m. \tag{2.15}$$

Equation (2.14) implies that W^+ inherits nonsingularity from W if σ_t is nonzero, and then conditions **(1)** and **(2)** are satisfied by the new interpolation points. The property $\sigma_t > 0$ can usually be achieved without difficulty, because σ_t is defined to be $\alpha_t \beta_t + \tau_t^2$, and α_t and β_t are both nonnegative in exact arithmetic. On iterations when $\underline{d}_k$ is a trust region step and $F(\underline{x}_k^* + \underline{d}_k)$ is calculated, the value of t is chosen in the following way.

All of the numbers σ_i, $i = 1, 2, \ldots, m$, are derived from expression (2.15), which requires only $\mathcal{O}(m^2)$ operations, because β_i does not depend on i. Then, in the case $F(\underline{x}_k^* + \underline{d}_k) < F(\underline{x}_k^*)$, equation (1.3) provides $\underline{x}_{k+1}^* = \underline{x}_k^* + \underline{d}_k$, and the index t is defined by the equation

$$w_t^2 \, |\sigma_t| \;=\; \max\{ w_i^2 \, |\sigma_i| : i = 1, 2, \ldots, m \}, \qquad (2.16)$$

where w_i is the weight

$$w_i \;=\; \left\{ \max \left[1,\; \|\underline{x}_i - \underline{x}_{k+1}^*\| \,/\, \max(0.1\Delta_{k+1}, \rho) \right] \right\}^3, \qquad i = 1, 2, \ldots, m. \qquad (2.17)$$

The purpose of the weights is to give some preference to the removal of an interpolation point that is far from $\underline{x}_{k+1}^*$. In the alternative case $F(\underline{x}_k^* + \underline{d}_k) \geq F(\underline{x}_k^*)$, equation (1.3) requires the interpolation point $\underline{x}_k^* = \underline{x}_{k+1}^*$ to be retained. Then the algorithm sets t to the integer in $[1, m]$ that maximizes $w_t^2 |\sigma_t|$ subject to $\underline{x}_t \neq \underline{x}_k^*$, but occasionally the current set of interpolation points is not altered, the change (1.18) being made when $w_t^2 |\sigma_t| > 1$ holds. A similar technique is employed by UOBYQA, in order to improve the conditioning of the system of interpolation equations.

Another task that is important in practice is the changes to the vector $\underline{x}_0$ of expression (2.9), in order to reduce the effects of rounding errors. The method that is employed by the current algorithm is crude and provisional. Specifically, if the k-th iteration applies the updating formula (2.14), it also tests the inequality $\|\underline{d}_k\| \leq 0.1 \|\underline{x}_{k+1}^* - \underline{x}_0\|$, the vectors $\underline{d}_k$ and $\underline{x}_{k+1}^*$ being available. Then $\underline{x}_0$ is moved to $\underline{x}_{k+1}^*$ if and only if this inequality holds. The resultant modification of H is addressed in Section 5 of Powell (2002b).

3. Numerical results and conclusions

The author had intended to present numerical results for several test problems at the Hangzhou conference in December, 2002, but, as mentioned already, the development of the algorithm of Section 2 is not yet complete. Further, he switched from development to producing numerical results less than one week before leaving Cambridge for that conference. In the development work the number of variables n was only three, but in the experiments the range of n is from $n = 10$ to $n = 160$. Fortunately, most of the numbers of iterations and all the final accuracies remained satisfactory when n was increased in this way, so no modifications were made in December to the current version of the algorithm. The development work with $n=3$ has been resumed since the conference, however, and it has caused a few recent changes, which are included in the details of Section 2. Therefore, because the results of this section were calculated by the newer version of the algorithm, the entries in the table below are different from the similar numerical results that were presented in Hangzhou. These differences are unimportant to the main conclusion of this section.

Only one test problem with more than three variables has been tried so far, because the author is still determined to reduce the damage from computer rounding errors in difficult cases with $n=3$. This test problem has also been used to demonstrate some features of the UOBYQA software (Powell, 2002a). Its objective function has the form

$$F(\underline{x}) \;=\; \sum_{i=1}^{2n} \Big\{ b_i - \sum_{j=1}^{n} \Big(S_{ij} \sin(\theta_j x_j) + C_{ij} \cos(\theta_j x_j) \Big) \Big\}^2 , \qquad \underline{x} \in \mathcal{R}^n, \tag{3.1}$$

where b_i, S_{ij}, C_{ij} and θ_j are parameters whose values depend on random numbers. Specifically, each element of the $2n \times n$ matrices S and C is picked independently from the uniform distribution of random integers on $[-100, 100]$, each scaling factor θ_j, $j=1,2,\ldots,n$, is from the logarithmic distribution of random numbers on $[0.1, 1]$, and the parameters b_i, $i=1,2,\ldots,2n$, are defined by the condition $F(\underline{x}^*)=0$, where $\underline{x}^*$ has the components $x_j^* = \hat{x}_j^*/\theta_j$, $j=1,2,\ldots,n$, the numerators $\hat{x}_j^*$, $j=1,2,\ldots,n$, being picked independently from the uniform distribution on $[-\pi, \pi]$. Thus $F(\underline{x})$, $\underline{x} \in \mathcal{R}^n$, is bounded and periodic, with a least value of zero, which occurs at $\underline{x} = \underline{x}^*$. The presence of $2n$ terms in the sum of squares (3.1) causes $\nabla^2 F(\underline{x}^*)$ to be well-conditioned in general, the condition number being about 100, because of the factors θ_j, $j=1,2,\ldots,n$. The initial vector of variables, namely $\underline{x}_0^*$, is given the components $(\hat{x}_j^* + 0.1 \hat{y}_j^*)/\theta_j$, $j=1,2,\ldots,n$, where each $\hat{y}_j^*$ is also from the uniform distribution on $[-\pi, \pi]$. Further, the values of $\rho_{\rm beg}$ and $\rho_{\rm end}$ are set to 10^{-1} and 10^{-6}, respectively, and the choices of m and n are stated below.

The main purpose of the numerical experiments is to investigate the dependence on m and n of the total numbers of function evaluations that occur. Therefore we let n take the values 10, 20, 40, 80 and 160, and we compare $m=2n+1$ with $m=\frac{1}{2}(n+1)(n+2)$, because the first choice requires much less work per iteration, and in the second case the interpolation conditions take up all the freedom in every quadratic model, which may reduce the number of iterations. Further, the intermediate value of m in Table 1 is about the geometric mean of the other two values, because this choice may be suitable when the work per iteration is $\mathcal{O}(n^3)$. The largest m is dropped for $n=80$, however, and only the smallest m is tried with $n=160$, in order to avoid very long computations. We generate five objective functions of the form (3.1) for each n by choosing different random numbers, and also one starting point $\underline{x}_0^*$ is picked automatically for each F. Thus the same five

m	n	Counts of F values	Final values of F	Times in seconds
21	10	313 – 469	$3\times d^{-10}$ – $2\times d^{-9}$	0.34 – 0.51
37	10	301 – 324	$1\times d^{-11}$ – $1\times d^{-9}$	0.55 – 0.60
66	10	214 – 279	$2\times d^{-14}$ – $3\times d^{-11}$	0.69 – 0.87
41	20	992 – 1155	$2\times d^{-9}$ – $4\times d^{-9}$	5.1 – 5.9
97	20	909 – 1133	$2\times d^{-10}$ – $8\times d^{-10}$	10.1 – 13.0
231	20	616 – 855	$7\times d^{-13}$ – $2\times d^{-11}$	19.9 – 27.5
81	40	2062 – 2469	$9\times d^{-9}$ – $3\times d^{-8}$	55 – 66
264	40	2342 – 2935	$2\times d^{-9}$ – $1\times d^{-8}$	230 – 289
861	40	3205 – 11335	$1\times d^{-10}$ – $9\times d^{-10}$	1797 – 7094
161	80	4173 – 4835	$3\times d^{-8}$ – $5\times d^{-8}$	727 – 839
731	80	4739 – 5825	$6\times d^{-9}$ – $2\times d^{-8}$	4877 – 6043
321	160	6697 – 8208	$9\times d^{-8}$ – $3\times d^{-6}$	8465 – 10164

Table 1: Numerical results for objective functions of the form (3.1)

minimization problems are solved for the different values of m, which provides five minimization calculations for each row of Table 1. The headings of the table state the quantities that are recorded for each of these problems, and the main part of the table gives the least and greatest values of these quantities for the five calculations that have been mentioned. Here a "Counts of F values" figure is the total number of evaluations of F by the algorithm in one experiment, the notation $\alpha\times d^{-\beta}$ stands for $\alpha\times 10^{-\beta}$, and every number in the last column is the total computation time in seconds for the work of one application of the algorithm. All the calculations were run in Fortran on a Sun Ultra 10 workstation.

The entries in the $n=10$ and $n=20$ rows of Table 1 are not very interesting. Indeed, the third column suggests that an increase in m reduces the number of evaluations of the objective function, the fourth column shows that good accuracy is achieved, a typical value of F for random variables being about $n^2\times 10^4$, and the timings in the last column indicate that smaller values of m are more efficient if the calculation of $F(\underline{x})$ for any $\underline{x}$ requires only $\mathcal{O}(n^2)$ operations. A remarkable feature of the $n=40$ rows, however, is that not only the least computation times but also the fewest numbers of function evaluations occur for the smallest choice of m. This observation receives further attention below. We note also that, for $n=40$ and $m=861$, the range 3205–11335 of the "Counts of F values" is huge. This remark is embarrassing to the author, because he has found that these figures are highly sensitive to changes in the Fortran implementation of the algorithm, even when the changes would not alter the results in exact arithmetic, which provides the main reason for the current research on reducing the damage from computer rounding errors. The UOBYQA algorithm is analogous to our method with $m=\frac{1}{2}(n+1)(n+2)$, but, when it was applied to five cases of the test problem of this section with $n=40$, in order to calculate one of the results in Table 4 of Powell (2003), the total numbers of function values ranged from 1970 to only 2275.

The ranges of the "Counts of F values" in the other rows of our table are so benign, however, that their contributions from computer rounding errors may be negligible in comparison to the variances that arise from the randomness in the construction of F. We can also take the view that, if rounding errors have a strong influence on the numbers of iterations, then an increase in the amount of work is more likely than a decrease. Therefore the given results suggest that, when $m=2n+1$ is chosen and n is large, then the solution of our test problem requires at most about $60n$ values of F. Further, regardless of any assumptions, we see that five unconstrained minimization calculations with 160 variables have been solved, and that in each case the objective function was calculated no more than 8208 times. On the other hand, UOBYQA would require 13041 values of F to generate its first quadratic model, because a quadratic function of 160 variables has this number of independent parameters. Therefore, although the algorithm is not in its final form, we can conclude already that, instead of seeking accuracy in the approximation $\nabla^2 Q_k \approx \nabla^2 F$, it may be much more efficient to update $\nabla^2 Q_k$ by the least Frobenius norm technique of this paper.

Our discussion of the theorem in Section 1 also suggests that fewer than $\mathcal{O}(n^2)$ values of the objective function may be sufficient to achieve high precision in unconstrained calculations without derivatives. Indeed, we recall that the limit (1.13), which is due to equation (1.11), does not require $\|\nabla^2 Q_k - \nabla^2 F\|_F$ to tend to zero. Therefore, when convergence properties are deduced from the limit, the number of parameters of a quadratic model is irrelevant. Further, if n is huge, then the value of n itself may also be irrelevant, except for the construction of the first quadratic model. It follows from these remarks that it may become possible to minimize functions of hundreds of variables to high accuracy without the calculation of any derivatives.

Acknowledgements

This paper was written at the University of Minnesota, where the author participated in the IMA Program on Optimization for two months, and was a Senior Ordway Visitor in the School of Mathematics. He is very grateful for the excellent facilities, hospitality and support that he received. He also thanks John Dennis for several helpful comments on the first draft of this work. The remarks of a referee provided further improvements.

References

C.G. Broyden, J.E. Dennis and J.J. Moré (1973), "On the local and superlinear convergence of quasi-Newton methods", *J. Inst. Math. Appl.*, Vol. 12, pp. 223–245.

A.R. Conn, N.I.M. Gould and Ph.L. Toint (2000), *Trust-Region Methods*, MPS–SIAM Series on Optimization (Philadelphia).

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

R. Fletcher (1987), *Practical Methods of Optimization*, John Wiley & Sons (Chichester).

J.J. Moré and D.C. Sorensen (1983), "Computing a trust region step", *SIAM J. Sci. Stat. Comput.*, Vol. 4, pp. 553–572.

M.J.D. Powell (2002a), "UOBYQA: unconstrained optimization by quadratic approximation", *Math. Programming*, Vol. 92, pp. 555–582.

M.J.D. Powell (2002b), "Least Frobenius norm updating of quadratic models that satisfy interpolation conditions", Report No. DAMTP 2002/NA08, University of Cambridge (to be published in *Math. Programming*).

M.J.D. Powell (2003), "On trust region methods for unconstrained minimization without derivatives", *Math. Programming*, Vol. 97, pp. 605–623.