A MODIFIED LEAST SQUARES ALGORITHM FOR SOLVING SPARSE N x N SETS OF NONLINEAR EQUATIONS

by

Arthur W. Westerberg & Stephen W. Director

DRC-06-5-79

January 1979

* Department of Chemical Engineering
** Department of Electrical Engineering
Carnegie-MelIon University
Pittsburgh, PA 15213

This work was supported by AFOSR Grant No. 74-2605A
and NSF Grants ENG-76-80149 and ENG-75-23078.

Abstract

In this paper we describe a algorithm for solving sparse n x n sets of nonlinear algebraic equations. This algorithm is like the Levenberg-Marquardt algorithm in that at each iteration the step size taken affects the direction selected to search; this direction lies between the Newton and gradient directions. Unlike the Levenberg-Marquardt schemes the sparsity of the original equations is preserved and thus sparse matrix methods can be employed for solving the linearized system of equations.

Scope

This paper presents a detailed description of an algorithm suitable for solving large sparse sets of nonlinear algebraic equations. The approach is closely related to the Newton-Raphson (N-R) method, a computational method which is becoming quite popular in the recent Chemical Engineering literature for solving large flowsheeting and related problems. The N-R method linearizes the equations about the current solution point and solves for the root of the linearized equations to use as the next guess for the solution. This method sometimes suffers because the predicted step may result in an estimate of the solution which is poorer than the old estimate guess. A common strategy used to try and avoid this problem is simply to take a smaller step in the same N-R direction. However, numerous examples exist for which this strategy fails.

An alternate procedure for solving nonlinear equations which overcomes this convergence problem was developed by Levenberg (1944) and Marquardt (1963). This least squares based method moves the search direction away from the N-R direction and toward the steepest descent direction of the function $S(x) = \sum_i f_i^2(x)$ if a shorter step size is desired. The L-M method has numerical drawbacks if the equations are nearly singular and usually results in a much denser set of equations to solve than the N-R method. Furthermore the step size is controlled only indirectly or else a major computational effort is required.

We have developed an algorithm for solving large sparse sets of nonlinear equations which has the convergence advantages of the L-M method but avoids the numerical disadvantages, i.e., the extra fill in. In addition the step size can be controlled exactly and with little effort. The algorithm was motivated by the dog-leg algorithm of Powell [1970] .

Conclusions and Significance

An algorithm is presented which is specifically designed to solve large sets of nonlinear algebraic equations. The step size used at each iteration is controlled as suggested by Reid [1972] to prevent excursions far beyond where the current linearization is valid. The convergence results of the algorithm on several known test problems is excellent.

1. Introduction

The classical approach for solving n nonlinear equations in n variables

$$f(x) = 0 \tag{1}$$

is the Newton-Raphson iteration method. In the most commonly used modification to this method the (k+1)st iterate for x is given by

$$x^{(k+1)} = x^{(k)} + a\, \eta^{(k)} \tag{2}$$

where $\eta^{(k)}$ is the Newton vector

$$\eta^{(k)} = - [J^{(k)}]^{-1} f(x^{(k)}) = - [J^{(k)}]^{-1} f^{(k)} , \tag{3}$$

and

$J^{(k)}$ is the Jacobian matrix of first partial derivates

$$J^{(k)} = \left| \frac{\partial f_i}{\partial x_j} \right|_{x^{(k)}} . \tag{4}$$

'a' is a scalar which is selected to insure that

$$S(x^{(k+1)}) < S(x^{(k)}) \tag{5}$$

where

$$S(x) = \sum_{i=1}^{n} f_i^2(x) = f^T(x)\, f(x) . \tag{6}$$

Powell [1970] has presented an example where this algorithm fails by converging to a point $\hat{x}$ which is clearly not the solution and where a step is readily found which will reduce $S(\hat{x})$. Moreover, the failure is not caused by inexact arithmetic.

The usual measure of success for locating a solution to equations (1) is to find an x which reduces S(x) in (6) to an acceptably small value.

$$\min_x S(x) = f(x)^T f(x)$$

The classical iteration here is a Gaussian step (Powell [I965])which corresponds to finding g^{k} at each step that satisfies

$$\min_I \{A_k(\S)\} \qquad \text{(PI)}$$

where

$$\backslash(e) - \%||f(x^{k}) + j^{(k)}e||^2 = \%(f(x^{(k)}) + j^{(k)}e)^T(f(x^{(k)})_+ j^{k}e)$$

Thus

$$\S(k) = x(k+D \cdot x(k) = \_ (J(k)^T j(k))\text{-}i\, j(k)^T f(k) \qquad (7)$$

(k)

which reduces to a Newton step if J^{v} ' is n x n.

Selecting the parameter $^f a^f$ in (2) suggests we usually wish to take a smaller step than a full Newton step if we are far from the solution. The Newton step is based on a local linearization which is likely to be valid only in the immediate vicinity of x. There is no £ priori reason to assume the Newton direction *1*\ is the best if we are going to limit our step size. Levenberg [1944] and Marquardt [1963] thus

proposed finding the step ^v ' which solves the problem

$$\min_{\mu} \{\phi_k(\mu) \mid ||\mu||^2 = \mu^T \mu \leq \delta^2\} \qquad \text{(P2)}$$

The Levenberg-Marquardt step is found by solving the linear system of equations

$$(J^{(k)T} J^{(k)} + X I)\mu^{(k)} = - J^{(k)T} f^{(k)} \qquad (8)$$

$$x^{(k+1)} = x(k) + \mu(k)$$

where $X \geq 0$ is selected such that

$$\lambda = \begin{cases} 0 & \text{if } ||n^{(k)}||^2 < \alpha^2 \\ \geq 0 & \text{otherwise} \end{cases}$$

In principle X should be selected such that $||y(X)||^2 = \delta^2$ if $||n^{(k)}||^2 \geq \delta^2$.

Marquardt [1963] noted in his papier that the direction p, $= TJ^{k}\Lambda$ if X = 0 and it moves towards the direction of steepest descent, $y^{(k)}$ > where

$$\gamma^{(k)} = - \frac{1}{2} \left(\frac{\partial S}{\partial x}\right)_{x^{(k)}} = -J(k)^T f(k) \qquad (9)$$

as X -> °°. Also the vector length is a continuous decreasing function of X, tending to a length of zero as the vector moves toward the direc--

tion of steepest descent. Powell [1970] developed his so-called "dog-leg" algorithm based on this observation. The dog-leg algorithm was developed to solve n equations in n unknowns and involves finding $Tp^{Ck,)}$ and $y^{(k)}$. Powell's step, $n^{(k)}$, is the Newton step $Tl^{(k)}$ if $||T]^{(k)}|!^2 \wedge f>^2$. Otherwise it is along $y^{(10}$ until a minimum in that direction is predicted in fJ(TT) . Then it is along the straight line joining this minimum point in the $y^{(k)}$ direction to the tip of $Tp^{(10}$ ' . This path, along $y^{v(10}$ ' to the minimum point and then on a straight line to the end of $T|^{v(10}$, is a dog-leg path.

The dog-leg algorithm is computationally more attractive than the Levenberg-Marquardt algorithm for several reasons. First the Leveriberg-Marquardt algorithm can have difficulties. If $J^{(10}$ is illconditioned with condition number p, then $3^{(k)T}3^{(k)}$ is much more illconditioned with condition number $p^{\wedge}$ (Steinberg [1974]). When the Levenberg-Marquardt iteration approaches convergence, X tends to zero and the coefficient matrix in (8) thus becomes potentially illconditioned. Also, if $3^{(k)}$ is a sparse matrix, the matrix $(J^{v\sim'}3^{\cdots'} + X\ I)$ will usually be dense thus precluding the use of sparse matrix methods for solving (8). In contrast, for the dog-leg algorithm, one needs to find the Newton direction $Tp^{(k)}$ (which corresponds to solving a set of sparse linear equations with $3^{(W}$ being the coefficient matrix) and the gradient direction $y^{(k)}$ (which involves using 3^{IOO} only in a matrix-vector multiplication) . One thus avoids dealing with a matrix denser and/or more illconditioned than $J^{(k)}$.

Other investigators (for example, Steen and Byrne [1973] and Jones [1970]) have provided algorithms for solving the general least squares problem where the permitted steps are restricted in length and must lie in the subspace spanned by the Gaussian step $\S^{(k)}$ and the steepest descent direction $y^{x(k)}$. Our main emphasis here is to avoid dealing directly with the coefficient matrix $J^{T}J$,

which Powell did and we do because we limit our attention to the case when the Jacobian matrix J is n x n. By a similar limitation, these algorithms could also deal with $T_p^{(It)}$ rather than $\S^{(k)}$. The principal difference is that we find the direction for the specified step size which should yield the greatest decrease in 0. As with other methods this direction is based on a local linearization about the current value of x.

In this paper we present an algorithm which has the computational attractiveness of the dog-leg algorithm combined with the convergence properties of the Levenberg.-Marquardt algorithm. The paper is divided as follows. In Section 2 we develop the equations needed for selecting the iteration step given a prescribed step size. The step is calculated without the introduction of the computational difficulties possible in the Levenberg-Marquardt algorithm. Section 3 proves the step selected has similar characteristics to the Marquardt step. Section 4 gives a complete algorithm, where the step size is updated at each iteration to reflect one's confidence in the accuracy of the linearization of f(x) about the current point $x^{(k)}$ and is identical to the approach in Reid [1972]. Section 5 presents several numerical examples.

2. The Iteration Step

Motivated by the best qualities of both Levenberg-Marquardt algorithm and Powell's "dog-leg" algorithm, we wish to select a step of prescribed maximum length but which lies in the subspace spanned by the gradient vector, $-2y>$ and the Newton vector, $T|^{(k)}$. The step U) will then be the step u> which solves the following problem.

$$\text{Min } \{\phi(\omega) \mid \omega^T\omega \le \delta^2,\ \omega = \alpha_1\eta^{(k)} + \alpha_2\gamma^{(k)}\} \qquad \text{(P3)}$$

We replace <u directly by

$$u = [\eta^{(k)},\ \gamma^{(k)}]\alpha = G\alpha \qquad (10)$$

and write the Lagrange function

$$L(\alpha,\lambda) = \tfrac{1}{2}(f(x^{(k)}) + J^{(k)}G\alpha)^T(f(x^{(k)}) + J^{(k)}G\alpha)$$

$$+ \lambda[(\alpha^T G^T G\alpha - \delta^2)] \qquad (11)$$

The solution to (P3) is at a stationary point of L(a,X); we set $\frac{\partial L}{\partial \alpha} = 0$ and obtain

$$(G^T J^{(k)T} J^{(k)} G + \lambda I)G\omega = -G^T J^{(k)T} f(x^{(k)}) \qquad (12)$$

or

$$\left\{\begin{bmatrix} \eta^{(k)T} J^{(k)T} J^{(k)} \eta^{(k)} & \eta^{(k)T} J^{(k)T} J^{(k)} \gamma^{(k)} \\ \gamma^{(k)T} J^{(k)T} J^{(k)} \eta^{(k)} & \gamma^{(k)T} J^{(k)T} J^{(k)} \gamma^{(k)} \end{bmatrix} + \begin{bmatrix} \eta^{(k)T}\eta^{(k)} & \eta^{(k)T}\gamma^{(k)} \\ \gamma^{(k)T}\eta^{(k)} & \gamma^{(k)T}\gamma^{(k)} \end{bmatrix}\lambda\right\}$$

$$\times \begin{bmatrix} \alpha_1 \\ \alpha_2 \end{bmatrix} = -\begin{bmatrix} \eta^{(k)T} \\ \gamma^{(k)T} \end{bmatrix} J^{(k)T} f^{(k)} \qquad X \ge 0 \qquad (13)$$

It is convenient to define

$$p^{(k)} = -J^{(k)T}y^{(k)} \qquad (14)$$

and note the relationships

$$_{n}(k)^{T}{}_{y}(k) = {}_{f}(k)^{T}{}_{f}(k) \tag{15}$$

$$_{f}(k)^{T}{}_{B}(k) = {}_{Y}(k)^{T}{}_{y}(k) \tag{16}$$

Equation (13) becomes

$$\begin{bmatrix} \|f^{(k)}\|^2 & |,_{Y}(k)|,^2 \\ MY^{00}\|^2 & \|_{B}<k),,2 \end{bmatrix} + \begin{bmatrix} \|n^{(W}|\,|2 & \|f^{(k)}|\,|^2 \\ JU^{(k)}|\,|^2 & \|Y^{(k)}|\,|^2 \end{bmatrix} X \begin{bmatrix} \alpha_{\dot{x}} \end{bmatrix} = \begin{bmatrix} Mf^{(k)}i\,|^2 \\ JIY^{(k)}|\,|^2 \end{bmatrix} \tag{17}$$

Given a value for X, $X \geq 0$, we can find α_i and c_u by first finding the vectors $f^{(k)}$, $y^{(k)}$ (equation (9)), $r/^{k)}$ (equation (3)) and $3^{(k)}$ (equation (14)). $y^{(k)}$ is calculated by a matrix-vector multiplication involving $J^{(k)^T}$, $r^{(k)}$ is obtained from the solution of a set of linear equations using $J^{(k)}$ as the coefficient matrix and $3^{(k)}$ is found by a matrix-vector multiplication involving $J^{(k)}$.

The final value of X can be selected such that $||or^{(k}\hat{}|| = 6$ if $IIT/\hat{}M > 6$. Otherwise $X = 0$ is selected and $u/^{k}\hat{} = r/\hat{}$ results.

It should be noted that (11) (alternatively (17)) can be written

$$(A + BX)a = e \tag{18}$$

where A, B and *e* are fixed. Thus finding X such that $|\,|u)^{(k)}|\,| = 6$ can be solved numerically (for example, using a secant-based method) as follows:

1. Guess X > 0.

2. Find α_1 and α_2 solving (18).

3. Evaluate ||u)||.

4. If ||C13| | ^ 6, adjust X and iterate from 2. Otherwise, exit.

Step 2 involves solving two linear equations in the unknowns α_1 and a_2 and is trivial to perform.

3. Properties of the Step CD

Assume problem (P3) is not singular; i.e., J has full rank for all x. We note initially then that $||Y^{(k)}||$t ll^ll and $l|P^{(k)}||$ are zero if and only if $||f\hat{}^{k}\hat{}||$ is zero because of their definitions. We shall also observe that A and B in (18) are symmetric and at least positive semi-definite because each is formed by the product of a matrix with its transpose. We now prove the following lemma and theorems.

Lemma 1: If and only if $Tp^{(k)}$ is colinear with $y^{(k)}$, then A and B are positive semi-definite. Otherwise they are positive definite.

Proof: Using the definitions of A and B in (12), the proof is obvious.

Theorem 1: Assume $||f\hat{}^{k}\hat{}||$ > 0. Let a(X) be a solution to (18) for a given value of X £ 0. Let

$$\omega(\lambda) = (n^{(k)}, Y^{(k)})a(X) \quad .$$

Then $6^{2} = ||a)(X)||^{2}$ is a continuous, decreasing function of X £ 0 such that as X -> «>, 6^{2} -* 0.

Proof: We consider two cases: Case 1) $n^{(k)}$ and $Y^{(k)}$ a^e colinear, Case 2) $n^{(k)}$ and $Y^{(k)}$ are not colinear.

Case 1) If $_{Y}\wedge^{k)}$ and *ffi*^ are colinear then

$$_{\gamma}00 = \cdot\ j(k)T_{\pounds}(k) = {}_{aT1}(k)\ m\ {}_{aJ}(k)\text{-}l_{f}(k)$$

or

$$(J^{<k>}\ J^{(k)^{T}} - a\ I)f^{(k)} = 0$$

where $f^{w} \wedge 0$ is an eigenvector of $R = J^{v}\ 'j^{v}\ '^{.T}$ and a is the corresponding eigenvalue, $a > 0$. We find then that

$$n^{T}n = f^{(k)^{T}}\ (J^{(k)^{T}})^{-1}\ (J^{(k)})^{-1}\ f^{(k)} = f^{(k)T}\ f^{(k)}/a$$

and similarly $_{Y}<^{k}>V^{k}> = a\ f\ «\ V^{k)}$, $fo = a^{2}\ f\ «\ V^{k}\ \backslash$ Thus (9) becomes

$$\left[\begin{pmatrix} 1 & a \\ a & a^{2} \end{pmatrix} + \lambda \begin{pmatrix} I/a & 1 \\ 1 & a \end{pmatrix}\right] \begin{pmatrix} {}_{a}l \\ \alpha_{2} \end{pmatrix} = \begin{pmatrix} 1 \\ a \end{pmatrix}$$

which are clearly two dependent equations with all solutions satisfying

$$({}^{Ct}l + a\ \alpha_{2}) = \frac{a}{a+\lambda}\ , \qquad a > 0\ , \qquad A \geq 0\ .$$

For this case

$$\delta^{2} = ||\omega(\lambda)||^{2} = ||(\alpha_{1} + a\ \alpha_{2})\eta^{(k)}||^{2} = \left(\frac{a}{a+\lambda}\right)^{2}\ ||\eta^{(k)}||^{2}$$

Thus 6^{2} is a continuous, decreasing function of X such that as X -*• °°, 6^{2}-> 0.

Case 2) $Y^{(k)}$ and $r\backslash^{(k)}$ are not colinear. A and B are symmetric and, by Lemma 1, positive definite. Thus we can find a (nonsingular) matrix M such that

$$M^T A M = D$$

$$M^T B M = I$$

where D is a positive definite diagonal matrix [10]• Equation (18) thus gives

$$a(X) = M(D + XI)^{-1} M^T 8$$

We note that

$$6^2 = |\,|o)|\,|^2 = a^T B a = a^T (M^T)^{-1} (M)""^{\wedge}$$

which, using the above, becomes

$$6^Z = e^T M(D + XI)^{-?z} M^T e$$

Letting $v = M^T e$ this becomes

$$\delta^2 = \sum_i \left(\frac{v_i}{d_i + A}\right)^2 , \quad u_i > 0 , \quad X \geq 0$$

which is a continuous, decreasing function of $A \geq 0$ such that as $X \to \infty$, $6^2 + Q.I11$

Theorem 2: Let ζ be the angle between ω and $\gamma^{(k)}$. Then ζ is a monotone, decreasing function of $\lambda \geq 0$ such that as $\lambda \to \infty$, $\zeta \to 0$.

Proof: The proof is exactly that given for Theorem (3) in Marquardt [4]. It will not be repeated here.

Clearly then, as with the Marquardt direction, the direction moves from the Newton direction $\eta^{(k)}$ to the steepest descent direction $\gamma^{(k)}$ as λ moves from zero to ∞. The step length decreases from $||\eta^{(k)}||$ to zero at the same time.

4. The Algorithm

The algorithm presented here is a merging of appropriate steps from the algorithms by Reid **[1972] and Powell [1970],** modified by **our** results.

1. Initialize 6, E $_{ln}$, E, . ,1 and p; estimate x$^{(o)}$; set
   ' small' big' max ^{v*} '
   i = j = k = 0.
2. Evaluate $f^{(k)} = f(x^{(k)})$, $S^{(k)} = ||f^{(k)}||^2 = f^{(k)}V^{k)}$ and set
   i - i + 1.
3. If $||f^{(k)}||^2 < E^2$ „ or i + pj > I , exit.
   ^{11} " small ^{VJ} max
4. If i + p(j+l) * I_{max}, exit.
5. Evaluate $J^{(k)}$ - (& \ . **Set** j - J + 1.
   x
6. Evaluate $_{Y}{}^{(k)} = - (J<^{k}>)^{T} f<^{k}>$ and $||_{Y}{}^{(k)}||^2$.
7. If $||f^{(k)}||^2 * E^2_{big}||_{Y}<^{k}>||^2$, exit.
8. Solve $J<^{k}V^{k)} = - f^{(k)}$ for n^{00}. Evaluate $||n^{(k)}||^2$.
9. If $||n^{(k)}||^2 \wedge 6^2$, set $co^{(k)} = n^{(k)}$ and set $6^2 = ||n^{(k)}||^2$ and go to step 13.
10. If $3^{(k)}$ not evaluated yet, evaluate $B^{(k)} = - J^{(k)}y^{(k)}$ and
    $llB^{<k)}H^2$.
11. A. Guess X > 0.
    B. Solve equations (17) for OL_{1}. and $a_{\_}\wedge$. If these equations are (essentially) singular, go to step 12.
    C. Evaluate a) = a ^ ^ + $a_2y^{(k)}$ and $||a)||^2$.
    D. If $||\infty|\ulcorner \wedge 6$, adjust X and repeat from step B, otherwise continue.
    E. Set $0)^{v}{}' = a)$ and go to step 13.

12. ($n^{(k)}$ and $y^{(k)}$ must be (essentially) colinear.) Set

$o)^{(k)}=6n^{(k)}/||n^{(k)}||$.

13. Set $x^{(k+1)} = x^{(k)} + \omega^{(k)}$.

14. Evaluate $f^{(k+1)}$, $S^{(k+1)} = ||f^{(k+1)}||^2$ and set i = i + 1.

15. If $||f^{(k+1)}||^2 < E^{\wedge}$ or $i + {}_{Pj} \gtrsim 1^{\wedge}$, exit.

16. If $||f^{(k+1)}||^2 > ||f^{(k)}||^2$, set 6 = 6/2 and repeat from step 10.

17. Predict change in S (called $AS^{(k)}_{pred}$) based on linear model.

A. $Af^{<k>} = J^{<k} V^{k}\backslash$

B. $b^{(k>} = (f^{(k>)^T} A f^{(k>}$,

C. $AS^{(k>}_{pred} = 2b^{(k)} + ||Af^{(k)}||^2$.

18. Evaluate ratio of actual change in S to predicted change in S:

A. $AS^{<k>}_{act} = S^{(k+1>} - S^{(k)}$,

R r - $AS^{(k)}/A<?^{(k)}$

B_ r - $^{tf}act^{MS}pred$ *

19. If r > 1, reset r = 2 - r.

20. If r > 0.75, set 6 = 6-MIN(2,[0.25/(l-r)] $^{1/2}$ and go to step 22.

21. If r < 0.25, set 6 = 6/MAX(2,MIN0.Q, 2 + $AS^{(k}J/b^{(k)})$) .
act

22. Set k = k + 1 and repeat from step 4.

The number E^{small} is used to check for convergence in steps 3 and 15. E^{big} is used in a check in step 7 and causes the algorithm to exit if it appears the smallest step size possible to reduce the functions $f^{(k)}$ to zero will be much too large. This test was devised by Powell [1970], E_{big} should be set to the magnitude of a step in x which the user feels will move x too far from the vicinity of $x^{(o)}$.

Index i counts function evaluations and j counts Jacobian matrix evaluations, p is a factor which relates the relative effort required to evaluate the Jacobian matrix $J^{(k)}$ to evaluating the functions $f^{(k)}$. If T is the number of nonzero elements in J and n the number of functions, p might be set to x/n. If J is evaluated numerically, p could be set to the number of function evaluations which would be required (which could be considerably fewer than n (see Curtis, Powell and Reid [1974]). I_{max} is then used in a check in steps 3, 5 and 15 causing the algorithm to exit if the number of equivalent function evaluations exceeds I_{max}. The initial value of 6 bounds the initial step taken for x. It is adjusted by the algorithm and should probably be estimated on the high side, particularly if the evaluation of $J^{(k)}$ in step (4) is time consuming relative to the evaluation of $f^{(k)}$.

Neglecting the test limiting the number of function evaluations, this algorithm uses the same tests for terminating as Powell [1970]. Powell also restricts his step $TT^{(k)}$ to be in the subspace defined by $n^{(k)}$ and $Y^{(k)}$. Thus his theorems on convergence will hold for our algorithm too. We simply state his theorem 2 as applied to our algorithm.

Theorem 3 (Theorem 2—Powell): If the functions f(x) have continuous, bounded, first derivatives, and if the algorithm in this paper is applied to solve the system of nonlinear equations f(x) = 0, then the algorithm will finish after a finite number of iterations, due either to the test in Step 3, Step 15 or the one in Step 7 being satisfied.

5. Examples

In order to illustrate the effectiveness of the algorithm described in Section 4, we apply it to the following well-known test problems:

1) Rosenbrock's banana shaped valley (Rosenbrock [I960]):

$$f_{\pm} = 10(x_2 - x_1^2) = 0$$

$$f_2 - 1 - x_1 = 0$$

a) with the initial starting point $\underset{\sim}{x}^0 = (-1.2,1.0)$ and

b) with the initial starting point $\underset{\sim}{x}^0 = (-0.86,1.14)$. The solution is $\underset{\sim}{x} = (1,1)$.

2) Powell's quartic function (Powell [1962]):

$$f_1 = x_1 + 10x_2$$

$$f_2 = \sqrt{5}\,(x_3 - x_4)$$

$$f_3 = (x_2 - 2x_3)^2$$

$$f_4 = \sqrt{10}(x_1 - x_4)^2$$

with the initial starting point $\underset{\sim}{x}^0 = (3,-1,0,1)$. The solution is $\underset{\sim}{x} = \underset{\sim}{0}$.

3) Powell's problem (Brown [1973]):

$$f_{\pm} = 10000\ x^{\wedge} - 1$$

$$f_2 = \exp(-x_1) + \exp(-x_2) - 1.0001$$

with the initial starting point $\underset{\sim}{x}^0 = (0,1)$. the solution is $x - (1099xlCf^4, 9.096)$.

4) Brown's almost linear function (More and Cosnard [1976]):

$$f_k(x) = x_k + \sum_{j=1}^{n} x_j - (n+1) \quad , \quad 1 \le k \le n-1 \quad ,$$

$$f_n(x) = \left(\prod_{j=1}^{n} Y \right) - 1$$

for n=10 with the initial starting point $\underset{\sim}{x}^0 = (.5, .5\_\_\_\_, .5)$ The solution is $\underset{\sim}{x} = <l_f l, ----_f l)$.

The results of application of the algorithm of Section 4 to this problem are summarized in Table 1. We feel that these results indicate that the algorithm is robust and represents, at least in many if not all cases, an improvement over the results of other reported algorithms because of its ability to achieve at each iteration a computed desired step size. We have omitted direct comparisons with results reported for other algorithms because of the myriad and sometimes inconsistent figures of merit given for these algorithms.

Note, in each of the above examples we have started the algorithm with a large desired step size so that the first iteration is a Newton step. If this step size is too large, the algorithm reduces it quickly.

Problem No.	U^2	No. Func. (f) Eval.	No. Jacobian (J) Eval.	No. Iterations
la	$< 10^{-12}$	9	6	5
lb	$< 10^{-12}$	21	13	12
2	$< 10^{-12}$	13	12	11
3	$< 10^{-10}$	50	43	42
4	$< ID^{-14}$	8	4	3

Table 1

6. References

Brown, K.M., "Computer Oriented Algorithms for Solving Systems of Simultaneous Nonlinear Algebraic Equations,[11] in Numerical Solution of Systems of Nonlinear Algebraic Equations, G.D. Byrne and C.A. Hall (eds.), Academic Press, New York (1973), pp. 281-348.

Curtis, A.R., M.J.D. Powell and J.K. Reid, "On the Estimation of Sparse Jacobian Matrices,[11] J. Inst. Math. Applies., Vol. 13 (1974), pp. 117-119.

Hildebrand, F.B., Methods of Applied Mathematics, Prentice-Hall, Englewood Cliffs, N.J. (1952), pp. 74-80,

Jones, A., "SPIRAL—A New Algorithm for Non-linear Parameter Estimation Using Least Squares," Comput. J., Vol. 13 (1970), pp. 301-308.

Levenberg, K., "A Method for the Solution of Certain Nonlinear Problems in Least Squares," Quart. Appl. Math., Vol. 2 (1944).

Marquardt, D.W., "An Algorithm for Least Squares Estimation of Nonlinear Parameters," J. Soc. Ind. Appl. Math., Vol. 11 (1963), pp. 431-441.

More, J.J. and M.Y. Cosnard, "On the Numerical Solution of Nonlinear Equations," Argonne Nat'l. Lab, Memorandum 286 (1976).

Powell, M.J.D., "An Iterative Method for Finding Stationary Values of a Function of Several Variables," Comput. J., Vol. 5 (1962), p. 147.

Powell, M.J.D,, "A Method for Minimizing a Sum of Squares of Nonlinear Functions without Calculating Derivatives," Comput. J., Vol. & (1965), pp. 303-307.

Powell, M.J.D., "A Hybrid Method for Nonlinear Equations," in Numerical Methods for Nonlinear Algebraic Equations, P. Rabinowitz (ed.), Gordon and Breach (1970), pp. 87-114.

Reid, J.K., "Fortran Subroutines for the Solution of Sparse Systems of Nonlinear Equations," Report No. R7293, AERE Harwell, Didcot, Berkshire, England (1972).

Reid, J.K., "Least Squares Solution of Sparse Systems of Nonlinear Equations by a Modified Marquardt Algorith," in Decomposition of Large-Scale Problems, D.M. Hinmelblau (ed.), American Elsevier, New York (1973), pp. 437-445.

Rosenbrock, H.H., "An Automatic Method for Finding the Greatest or Least Value of a Function," Comput. J., Vol. 3 (1960), p. 175.

Steen, N.M. and G.D. Byrne, "The Problem of Minimizing Nonlinear Functionals," in Numerical Solution of Systems of Nonlinear Algebraic Equations, G.D. Byrne and C.A. Hall (eds.), Academic Press, New York (1973), pp. 185-239.

Steinberg, D.I., Computational Matrix Algebra, McGraw-Hill, New York (1974) .