SCALING AND QUASI-NEWTON METHODS

Jorge R. Paloschi*
John D. Perkins**
Department of Chemical Engineering
Imperial College
London - SW7 - United Kingdom

ABSTRACT

Internal scaling procedures for optimizing the numerical conditioning of problems of the form $f(x) = 0$ solved by Quasi-Newton methods are proposed and tested on a standard set of mathematical examples.

* Presently at: PLAPIQUI - Universidad Nacional del Sur
12 de Octubre 1842 - C.C. 717
8000 Bahía Blanca - Argentina

** Presently at: University of Sydney
Department of Chemical Engineering
Sydney - NSW - Australia

1. INTRODUCTION

We will be concerned with the problem of solving a general nonlinear system of algebraic equations. Consider a function $f: R^n \to R^n$ and the problem

$$f(x) = 0 \tag{1.1.}$$

Methods for solving (1.1.) are usually iterative. They generate a sequence $\{x_k\}$, $\{x_k\} \in R^n$, which if the method converges satisfies

$$\lim_{k\to\infty} x_k = x_* \tag{1.2.}$$

$$f(x_*) = 0 \tag{1.3.}$$

If we use the notation $F'(x)$, for the Jacobian matrix of the function f evaluated at x, it was observed that if f is a linear function the solution x_* can be found as

$$x_* = x + p^N \tag{1.4.}$$

where

$$p^N = - F'(x)^{-1} f(x) \tag{1.5.}$$

provided $F'(x)^{-1}$ is well defined.

If f is not a linear function but x_k is a point sufficiently near x_*, we can expect that defining

$$x_{k+1} = x_k + p_k^N \tag{1.6.}$$

will provide a better approximation to x_*, i.e.

$$\| x_{k+1} - x_* \| \leq \| x_k - x_* \| \tag{1.7.}$$

The iteration process (1.6.) is the Newton method and thus p_k^N is called the Newton step.

This method has been the basis for many successful methods developed in the past.

The computation of $F'(x)$ is often very expensive or even impossible, therefore methods have been devised for avoiding it.
Instead of $F'(x_k)$ an approximation B_k to it has been used, giving then instead of (1.5.) and (1.6.)

$$p_k = - B_k^{-1} f(x_k) \tag{1.8.}$$

$$x_{k+1} = x_k + p_k \tag{1.9.}$$

We can divide the methods for solving (1.1.) into two classes according to the way in which the approximation B_k is obtained. The first class

comprises those in which B_k is obtained by a finite difference approximation to the real Jacobian $F'(x_k)$. We can mention in this class the discrete Newton's method (Ortega and Rheinboldt (1970)), Brown's (1966) method and Brent's (1973) method. The second class comprises methods in which B_k is updated each iteration using a formula

$$B_{k+1} = B_k + A_k \tag{1.10.}$$

and the matrix A_k is determined by the method. Methods have been proposed using an updating matrix A_k of rank one (Broyden (1965), Barnes (1965), Paloschi and Perkins (1982)) and also with rank greater than one (Schubert (1970)).
Numerical results obtained by Hiebert (1980), in testing general codes for solving (1.1.), show that the behaviour of the different codes is very dependent on the scale being used for solving the problem.
Consider the general change of scale

$$\hat{f}(\hat{x}) = D_f f(D_x^{-1} \hat{x}) \tag{1.11.}$$

An iterative method of the form (1.9.) will be said to be scale invariant if for a change of scale of the form (1.11.) it satisfies

$$\hat{x}_k = D_x x_k, \quad \forall k \tag{1.12.}$$

From all the methods available, only Newton's method and the methods presented by Paloschi and Perkins (1982) are scale invariant. However, due to the fact that finite precision will be used for real calculations, all methods, regardless of their theoretical properties, will be scale dependent (see the numerical results presented in Paloschi and Perkins (1982)).
This problem is directly related to the numerical conditioning of the methods. We will discuss here the relation between the numerical conditioning and the methods and propose a way for improving it.
We will first introduce the concept of condition number for systems of nonlinear algebraic equations, discuss its relation with methods of the form (1.8.) and (1.9.), and then propose a procedure for improving the numerical conditioning. Finally, we will test our proposals on a standard set of mathematical examples.

2. THE CONDITION NUMBER

The condition number has been introduced as a measure of the numerical conditioning for general matrices.

For a non-singular matrix $A \in L(R^n)$ the condition number $k(A)$ is defined as

$$k(A) = \| A \| \, \| A^{-1} \| \tag{2.1.}$$

for a given matrix norm $\| \ \|$.

If A defines a system of linear equations in R^n

$$A x = b \tag{2.2.}$$

it is a well known result (Ortega Rheinboldt (1970)) that if $B \in L(R^n)$ is close to A in the sense that

$$\| A^{-1} \| \, \| B - A \| < 1 \tag{2.3.}$$

then B is also non-singular, and for $b \neq 0$ the solutions x_* of (2.2.) and y_* of

$$B\, x = c \tag{2.4.}$$

satisfy the estimate (see Rheinboldt (1976))

$$\frac{\| x_* - y_* \|}{\| x_* \|} \leq \frac{k(A)}{1 - k(A) \, \| B-A \| / \| A \|} \left[\frac{\| B - A \|}{\| A \|} + \frac{\| b - c \|}{\| b \|} \right] \tag{2.5.}$$

As an example of the use of this number let us assume that in solving numerically the equation (2.2.) we have found an approximation y_* to x_* (the exact solution).

If $k(A)$ is a large number then the fact that $Ay_* \tilde{=} b$ does not mean that y_* is close to x_* ; we can deduce this using (2.5.) to obtain

$$\frac{\| x_* - y_* \|}{\| x_* \|} \leq k(A) \, \frac{\| b - Ay_* \|}{\| b \|}$$

In general we can say that the smaller $k(A)$, the better the result obtained in solving numerically (2.2.).

This important result has led to finding ways for transforming (2.2.) into an equivalent linear system having the same solution but smaller condition number.

This concept of condition number for linear systems has been generalized to systems of non-linear equations by Rheinbolt (1976) as follows:

For a given function $f: D \subset R^n \Rightarrow R^n$, closed set $C \subset D$ and point $z \in C$ define

$$\begin{aligned} \mu(f, C, z) &= \sup \{ t \in [0, \infty) \; ; \; \| f(x) - f(z) \| \geq t \, \| x - z \| \;\; , \; \forall x \in C \} \\ \nu(f, C, z) &= \inf \{ t \in [0, \infty) \; ; \; \| f(x) - f(z) \| \leq t \, \| x - z \| \;\; , \; \forall x \in C \} \end{aligned} \tag{2.6.}$$

and then, define the localized condition number

$$k(f, C, z) = \begin{cases} \dfrac{\nu(f, C, z)}{\mu(f, C, z)} & \text{if } \; 0 < \mu(f, C, z) \; , \; \nu(f, C, z) < \infty \\ \infty & \text{otherwise} \end{cases} \tag{2.7.}$$

It can be shown that (2.7.) reduces to (2.5.) if f is a linear function.

Rheinboldt (1974) has shown that if f is a continuous function in D and if the Jacobian $F'(x)$ of f is nonsingular in D then for any $\varepsilon > 0$ there is a $\delta > 0$ such that if

$$C = \{x \in R^n , \| x - z\| \leq \delta\} \subset D \qquad (2.8.)$$

then

$$|\nu(f, C, z) - \| F'(z)\| | \leq \varepsilon \qquad (2.9.)$$

$$|\mu(f, C, z) - \| F'(z)^{-1}\|^{-1} | \leq \varepsilon \qquad (2.10.)$$

and then, asymptotically near z, the conditioning of the non-linear function f and its Jacobian $F'(z)$ are the same.

An equivalent formula to (2.5.) is obtained for the non-linear case and then the condition number for systems of non-linear equations plays a similar role as it does for linear systems.

As it is done for solving linear systems, in dealing numerically with the problem (1.1.) one should try to solve a system for which its condition number is small.

3. METHODS FOR SOLVING $f(x) = 0$ AND THE CONDITION NUMBER

We are interested in methods of the form (1.8.) and (1.9.) for solving the problem (1.1.). Those methods can be represented with the following algorithm:

ALGORITHM 1

1 - Given X_o , B_o , ε

2 - Set $k = 0$

3 - If $\| f(x_k)\| \leq \varepsilon$ then stop

4 - $p_k = - B_k^{-1} f(x_k)$

5 - $x_{k+1} = x_k + p_k$

6 - Obtain B_{k+1} (this is determined by the particular method).

7 - $k = k+1$

8 - go to 3

The numerical performance of Algorithm 1 is affected by the condition number in two different ways.

a) The conditioning of the problem (1.1.) itself, as explained in section 2.

b) The conditioning of B_k since step 4 of Algorithm 1 implies solving the linear system.

$$B_k \, p_k = - f(x_k) \qquad (3.1.)$$

We will show that it is possible to reduce the condition number of B_k by using an internal scaling procedure. In addition, if B_k is close to the real Jacobian $F'(x_k)$ (which is not necessarily true, even when convergence is achieved, see Dennis and More (1977)) then a discussion of section 2 shows that in this case we will also be improving the condition number of the problem itself.

We will now define a property for methods implemented using Algorithm 1

which will be the basis for our results.

For a change of scale of the form (1.11.) define

PROPERTY S: $\hat{B}_k = D_f \, B_k \, D_x^{-1}$ (3.2.)

If a method satisfies property S then it means that we can obtain the approximation B_k for the Jacobian in the new scale just by multiplying the approximation in the original scale with the scaling matrices.

It has been shown by Paloschi and Perkins (1982) that Property S is a sufficient condition for a method being scale invariant for changes of the form (1.11.).

For a method satisfying property S it will be very easy to apply a change of scale of the form (1.11.) and since in general

$$K (D_f \, B_k \, D_x^{-1}) \neq K (B_k) \tag{3.3.}$$

we could try to obtain matrices D_f and D_x such that

$$K (D_f \, B_k \, D_x^{-1}) \leq K (B_k) \tag{3.4.}$$

In the following theorem due to Bauer (1963) we will find the theoretical basis for our choice of D_f and D_x.

THEOREM 3.1.: For a nonsingular matrix $A \in L(R^n)$ and nonsingular diagonal matrices D_1 and D_2, using the maximum norm for matrices,

$$\min_{D_1} K (D_1 A) \tag{3.5.}$$

and

$$\min_{D_2} K (A D_2) \tag{3.6.}$$

are achieved for D_1 and D_2 determined from

$$|A^{-1}| \, e = D_2 \, e \tag{3.7.}$$

$$|A| \, e = D_1^{-1} \, e \tag{3.8.}$$

($|A|$ means the original matrix with all its elements taken in absolute value, while e is a vector of all ones).

We can then see that it is possible to minimize, in some sense, the condition number of the approximation by scaling either the variables or the function. Conditions for achieving the same by scaling simultaneously the variables and the function could be obtained but since it is required to evaluate eigen-values for determining D_f and D_x it becomes a very costly procedure (see Bauer (1963)).

All we need for applying these results is being able to obtain B_k in the new scale given it on the original one. If a method satisfies

property S then we can see that it is under the required conditions. In fact, it is only required for a method to satisfy property S with $D_x = I$ for scaling the function or with $D_f = I$ for scaling the variables. In either case $\hat{B}_k$ is obtained by multiplying the scaling matrix according to (3.2.).

Newton's method and the methods proposed by Paloschi and Perkins (1982) satisfy property S for all k . This means that we can optimize $k(B_k)$ at all iterations by using an internal scaling based on Theorem 3.1. Broyden's (1965) method only satisfy property S if $D_x = I$ (see Malathronas and Perkins (1980)) which means we can optimize $k(B_k)$ at all iterations by using only function scaling. If the initial approximation B_o is obtained by finite differences and all the components of x_o are away from the origin then B_o satisfy property S (see Paloschi (1982)). This means we can optimize $k(B_o)$ for any method in which B_o is obtained by finite differences by scaling the variables or the function.

4. NUMERICAL RESULTS

We will apply our results of section 3 to the method of Broyden (1965) and the scale invariant methods proposed by Paloschi and Perkins (1982). The implementation details used for the code can be found in Paloschi (1982).

The set of examples which will be used is the proposed by More, Garbow and Hillstrom (1978). This set was used by Hiebert (1980), Chen and Stadtherr (1981) and Paloschi and Perkins (1982). It consists of a general set of 54 mathematical problems and a set of 12 chemical equilibrium problems. Both sets are described in Appendix A.

For the general set of mathematical problems a diagonal matrix Smn is used for testing the behaviour of the codes under different scaling conditions. The diagonal elements of the matrix Smn are defined by

$$\log_{10} \sigma_{mi} = (m((2i-n-1)/(n-1))) \qquad (4.1.)$$

for $i = 1,2,\ldots,n$.

For creating a set with variables badly scaled we use

$$\hat{f}(\hat{x}) = f(S_{5n}x) \qquad (4.2.)$$

and for functions badly scaled

$$\hat{f}(x) = S_{5n} f(x) \qquad (4.3.)$$

The original set of 54 examples is used in its original scale and also with the problems modified as indicated by (4.2.) and (4.3.). This gives a total of 162 problems.

We present in Table 4.1. all the scaling possibilities we could choose. The entries in the table is the number we will use to refer to the corresponding scaling possibility.

All computations were performed on a CDC CYBER 174 in single precision.

	Scaling the function		
	Never	First Iteration	Always
Scaling the variables			
Never	1	2	3
First iteration	4	5	6
Always	7	8	9

Table 4.1.: Different scaling possibilities.

The scalings 7, 8 and 9 can not be applied to Broyden's method because it does not satisfy property S for $Df = I$ and $Dx \neq I$. With this exception for Broyden's method we will try all the scaling possibilities for all methods. When both scalings are used simultaneously, we will present numerical results showing the two possibilities, i.e. first scaling the variables, then the functions and viceversa.

We present our results, in terms of percentage of success, in Tables 4.2. and 4.3. for the mathematical problems and in Table 4.4. and 4.5. for the chemical equilibrium problems.

	Scaling possibilities								
Method	1	2	3	4	5	6	7	8	9
Broyden	73	76	76	73	74	78	--	--	--
SI2	73	80	81	73	83	84	74	81	54
SI3	75	77	75	74	77	75	72	76	51
SI4	75	75	77	78	77	77	76	77	49

Table 4.2.: Percentage of success scaling first the variables and then the functions (mathematical set of problems).

	Scaling possibilities								
Method	1	2	3	4	5	6	7	8	9
Broyden	73	76	76	73	76	78	--	--	--
SI2	73	80	81	73	82	81	74	80	59
SI3	75	77	75	74	77	77	72	75	52
SI4	75	75	77	78	76	76	76	75	49

Table 4.3.: Percentage of success scaling first the functions and then the variables (mathematical set of problems).

	Scaling possibilities								
Method	1	2	3	4	5	6	7	8	9
Broyden	83	75	100	75	75	92	--	--	--
SI2	67	50	83	75	58	67	75	67	58
SI3	67	75	83	75	83	92	67	75	75
SI4	67	67	75	67	75	92	75	83	58

Table 4.4.: Percentage of success scaling first the variables and then the functions (chemical equilibrium set of problems).

	Scaling possibilities								
Method	1	2	3	4	5	6	7	8	9
Broyden	83	75	100	75	75	92	--	--	--
SI2	67	50	83	75	50	67	75	50	58
SI3	67	75	83	75	75	83	67	75	67
SI4	67	67	75	67	58	83	75	67	58

Table 4.5.: Percentage of success scaling first the functions and then the variables (chemical equilibrium set of problems).

The analysis of the numerical results shows that there is practically not much difference with the order in which the scaling is performed. For the chemical equilibrium set of problems the difference is important but due to the small size of the set (12 problems) we can not consider this difference significant.

Regarding the scaling policy to be chosen the numerical results show that policies 5 and 6 are the best in general. In particular, policy 3 performs better for two methods in the chemical equilibrium set of problems but again we should discard this as significant due to the size of the set.

Regarding the amount of work involved in these scalings, it depends on the implementation chosen. We need to have available B_k for scaling the functions and B_k^{-1} for scaling the variables. We will always have available either B_k^{-1} or a factorization of B_k. In our implementation, an LU factorization of B_k is available, thus for us $O(n^2)$ operations are needed for scaling the functions and $O(n^3)$ for scaling the variables.

5. CONCLUSIONS

We have presented internal scaling procedures which are simple to implement and whose inclusion in an algorithm for solving systems of algebraic non-linear equations produce a considerable improvement in robustness.

REFERENCES

Barnes J.G.P. (1965) - "An algorithm for solving nonlinear equations based on the secant method". The Comp. Journal 8.

Bauer F.L. (1963) - "Optimally scaled matrices". Num. Math. 5.

Brent R.P. (1973) - "Some efficient algorithms for solving systems of nonlinear equations". SIAM J. on Num. Anal. 10.

Brown K.M. (1966) - "A quadratically convergent method for solving simultaneous nonlinear equations". Ph.D. Diss., Purdue Univ.

Broyden C.G. (1965) - "A class of methods for solving nonlinear simultaneous equations". Math. of Comp. 19.

Chen H.S., Stadtherr M.A. (1981) - "A modification of Powell's dogleg method for solving systems of nonlinear equations". Comp. and Chem. Eng. 5.

Dennis J.E., More J.J. (1977) - "Quasi-Newton methods, motivation and theory". SIAM Review 19.

Hiebert K.L. (1980) - "A comparison of software which solves systems of nonlinear equations". Sandia Lab. Report, SAND, 80-0181.

Malathronas J.P., Perkins J.D. (1980) - "Solution of design problems using Broyden's method in a sequential modular flowsheeting package". Paper at: CHEM PLANT 80, Heviy-Hungary.

More J.J., Garbow B.S., Hillstrom K.E. (1978) - "Testing unconstrained optimization software". Report Argonne Nat. Lab. -ANL-AMD-TM-324.

Ortega J.M., Rheinboldt W.C. (1970) - "Iterative solution of nonlinear equations in several variables". Academic Press.

Paloschi J.R. (1982) - "The numerical solution of nonlinear equations representing chemical processes". Ph.D. Thesis, Univ. of London.

Paloschi J.R., Perkins J.D. (1982) - "Sacel invariant Quasi-Newton methods". Imperial College Report.

Rheinboldt W.C. (1976) - "On measures of ill conditioning for nonlinear equations". Math. of Comp., 30, pp. 104-111.

Schubert L.K. (1970) - "Modification of a Quasi-Newton method for nonlinear equations with a sparse function". Math. of Comp. 23.