Hindawi Publishing Corporation
Journal of Applied Mathematics
Volume 2012, Article ID 348654, 9 pages
doi:10.1155/2012/348654

Research Article

A Two-Step Matrix-Free Secant Method for Solving Large-Scale Systems of Nonlinear Equations

M. Y. Waziri,[1,2] W. J. Leong,[1] and M. Mamat[3]

[1] *Department of Mathematics, Faculty of Science, University Putra Malaysia, 43400 Serdang, Malaysia*
[2] *Department of Mathematics, Faculty of Science, Bayero University Kano, Kano 2340, Nigeria*
[3] *Department of Mathematics, Faculty of Science and Technology, Malaysia Terengganu University, Kuala Lumpur 21030 Terengganu, Malaysia*

Correspondence should be addressed to M. Y. Waziri, mywaziri@gmail.com

Received 9 November 2011; Revised 11 January 2012; Accepted 16 January 2012

Academic Editor: Renat Zhdanov

We propose an approach to enhance the performance of a diagonal variant of secant method for solving large-scale systems of nonlinear equations. In this approach, we consider diagonal secant method using data from two preceding steps rather than a single step derived using weak secant equation to improve the updated approximate Jacobian in diagonal form. The numerical results verify that the proposed approach is a clear enhancement in numerical performance.

1. Introduction

Solving systems of nonlinear equations is becoming more essential in the analysis of complex problems in many research areas. The problem considered is to find the solution of nonlinear equations:

$$F(x) = 0, \tag{1.1}$$

where $F =: R^n \to R^n$ is continuously differentiable in an open neighborhood $\Phi \subset R^n$ of a solution $x^* = (x_1^*, \ldots, x_n^*)$ of the system (1.1). We assume that there exists x^* with $F(x^*) = 0$ and $F'(x^*) \neq 0$, where $F'(x_k)$ is the Jacobian of F at x_k for which it is assumed to be locally Lipschitz continuous at x^*. The prominent method for finding the solution to

(1.1) is the Newton's method which generates a sequence of iterates $\{x_k\}$ from a given initial guess x_0 via

$$x_{k+1} = x_k - \left(F'(x_k)\right)^{-1} F(x_k), \tag{1.2}$$

where $k = 0, 1, 2 \ldots$. However, Newton's method requires the computation of the matrix entailing the first-order derivatives of the systems. In practice, computations of some functions derivatives are quite costly, and sometimes they are not available or could not be done precisely. In this case, Newton's method cannot be used directly.

It is imperative to mention that some efforts have been already carried out in order to eliminate the well-known shortcomings of Newton's method for solving systems of nonlinear equations, particularly large-scale systems. These so-called revised Newton-type methods include Chord-Newton method, inexact Newton's method, quasi-Newton's method, and so forth (e.g., see [1–4]). On the other hand, most of these variants of Newton's method still have some shortcomings as Newton's counterpart. For example, Broyden's method [5] and Chord's Newton's method need to store an $n \times n$ matrix, and their floating points operations are $O(n^2)$.

To deal with these disadvantages, a diagonally Newton's method has been suggested by Leong et al. [6] by approximating the Jacobian matrix into nonsingular diagonal matrix and updated in every iteration. Incorporating this updating strategy, Leong et al. [6], based upon the weak secant equation of Denis and Wolkonicz [7], showed that their algorithm is appreciably cheaper than Newton's method and some of its variants. It is worth to report that, they employ a standard one-step two-point approach in Jacobian approximation, which is commonly used by most Newton-like methods. In contrast, this paper presents a new diagonal-updating Newton's method by extending the procedure of [6] and employs a two-step multipoint scheme to increase the accuracy of the approximation of the Jacobian. We organize the rest of this paper as follows. In the next section, we present the details of our method. Some numerical results are reported in Section 3. Finally, conclusions are made in Section 4.

2. Two-Step Diagonal Approximation

Here, we define our new diagonal variant of Newton's method which generates a sequence of points $\{x_k\}$ via

$$x_{k+1} = x_k - Q_k^{-1} F(x_k), \tag{2.1}$$

where Q_k is a diagonal approximation of the Jacobian matrix updated in each iteration. Our plan is to build a matrix Q_k through diagonal updating scheme such that Q_k is a good approximation of the Jacobian in some senses. For this purpose, we make use of an interpolating curve in the variable space to develop a weak secant equation which is derived initially by Dennis and Wolkowicz [7]. This is made possible by considering some of the most successful two-step methods (see [8–11] for more detail). Through integrating this two-step information, we can present an improved weak secant equation as follows:

$$(s_k - \alpha_k s_{k-1})^T Q_{k+1} (s_k - \alpha_k s_{k-1}) = (s_k - \alpha_k s_{k-1})^T (y_k - \alpha_k y_{k-1}). \tag{2.2}$$

By letting $\rho_k = s_k - \alpha_k s_{k-1}$ and $\mu_k = y_k - \alpha_k y_{k-1}$ in (2.2), we have

$$\rho_k^T Q_k \rho_k = \rho_k^T \mu_k. \tag{2.3}$$

Since we used information from the last two steps instead of one previous step in (2.2) and (2.3), consequently we require to build an interpolating quadratic curves $x(v)$ and $y(v)$, where $x(v)$ interpolates the last two preceding iterates x_{k-1} and x_k, and $y(v)$ interpolates the last two preceding function evaluation F_{k-1} and F_k (which are assumed to be available).

Using the approach introduced in [8], we can determine the value of α_k in (2.2) by computing the values of v_0, v_1, and v_2. If $v_2 = 0$, $\{v_j\}_{j=0}^2$ can be computed as follows:

$$\begin{aligned} -v_1 &= v_2 - v_1 \\ &= \|x(v_2) - x(v_1)\|_{Q_k} \\ &= \|x_{k+1} - x_k\|_{Q_k} \\ &= \|s_k\|_{Q_k} \\ &= \left(s_k^T Q_k s_k\right)^{1/2}, \end{aligned} \tag{2.4}$$

$$\begin{aligned} -v_0 &= v_2 - v_0 \\ &= \|x(v_2) - x(v_0)\|_{Q_k} \\ &= \|x_{k+1} - x_{k-1}\|_{Q_k} \\ &= \|s_k + s_{k-1}\|_{Q_k} \\ &= \left((s_k + s_{k-1})^T Q_k (s_k + s_{k-1})\right)^{1/2}. \end{aligned} \tag{2.5}$$

Let us define β by

$$\beta = \frac{v_2 - v_0}{v_1 - v_0}, \tag{2.6}$$

then ρ_k and μ_k are give as

$$\rho_k = s_k - \frac{\beta^2}{1 + 2\beta} s_{k-1}, \tag{2.7}$$

$$\mu_k = y_k - \frac{\beta^2}{1 + 2\beta} y_{k-1}, \tag{2.8}$$

that is, α in (2.4) is given by $\alpha = \beta^2/(1 + 2\beta)$.

Assume that Q_k is a nonsingular diagonal matrix, the updated version of Q_k, that is, Q_{k+1} is then defined by

$$Q_{k+1} = Q_k + \Lambda_k, \tag{2.9}$$

where Λ_k is the deviation between Q_k and Q_{k+1} which is also a diagonal matrix. To preserve accurate Jacobian approximation, we update Q_{k+1} in such a way that the following condition is satisfied:

$$\rho_k^T(Q_{k+1})\rho_k = \rho_k^T\mu_k. \tag{2.10}$$

We proceed by controlling the growth error of Λ_k through minimizing its size under the Frobenius norm, such that (2.10) holds. Consequently we consider the following problem:

$$\begin{aligned} \min_{\Lambda^{(i)}\in R} \quad & \frac{1}{2}\sum_{i=1}^{n}\left(\Lambda_k^{(i)}\right)^2 \\ \text{s.t} \quad & \sum_{i=1}^{n}\Lambda_k^{(i)}\left(\rho_k^{(i)}\right)^2 = \rho_k^T\mu_k - \rho_k^TQ_k\rho_k, \end{aligned} \tag{2.11}$$

where $\Lambda_k^{(i)}$; $i = 1,2,\ldots,n$ are the diagonal elements of Λ_k and $\rho_k^{(i)}$; $i = 1,2,\ldots,n$ are the components of vector ρ_k.

In view of the fact that, the objective function in (2.11) is convex, and the feasible set is also convex, then (2.11) has an unique solution. Hence its Lagrangian function can be expressed as follows:

$$L(\Lambda_k,\lambda) = \frac{1}{2}\sum_{i=1}^{n}\left(\Lambda_k^{(i)}\right)^2 + \lambda\left(\sum_{i=1}^{n}\Lambda_k^{(i)}\left(\rho_k^{(i)}\right)^2 - \rho_k^T\mu_k + \rho_k^TQ_k\rho_k\right), \tag{2.12}$$

where λ is the Lagrangian multiplier associated with the constraint.

Taking the partial derivative of (2.12) with respect to each component of Λ_k then set them equal to zero, multiply the relation by $(\rho_k^{(i)})^2$ and sum them all together, yields

$$\Lambda_k^{(i)} = -\lambda\left(\rho_k^{(i)}\right)^2, \tag{2.13}$$

$$\sum_{i=1}^{n}\Lambda_k^{(i)}\left(\rho_k^{(i)}\right)^2 = -\lambda\sum_{i=1}^{n}\left(\rho_k^{(i)}\right)^4, \quad \text{for each } i = 1,2,\ldots,n. \tag{2.14}$$

Using (2.14) and the constraint, we obtain after some simplifications

$$\lambda = -\frac{\rho_k^T\mu_k - \rho_k^TQ_k\rho_k}{\sum_{i=1}^{n}\left(\rho_k^{(i)}\right)^4}. \tag{2.15}$$

Next, by substituting (2.15) into (2.13) and performing some little algebra, we obtain

$$\Lambda_k^{(i)} = \frac{\rho_k^T \mu_k - \rho_k^T Q_k \rho_k}{\sum_{i=1}^{n} \left(\rho_k^{(i)}\right)^4} \left(\rho_k^{(i)}\right)^2, \quad \forall i = 1, 2, \ldots, n. \tag{2.16}$$

Letting $H_k = \operatorname{diag}((\rho_k^{(1)})^2, (\rho_k^{(2)})^2, \ldots, (\rho_k^{(n)})^2)$ and $\sum_{i=1}^{n} (\rho_k^{(i)})^4 = \operatorname{Tr}(H_k^2)$, where Tr is the trace operation, we can finally present the updating formula for Q as follows.

$$Q_{k+1} = Q_k + \frac{(\rho_k^T \mu_k - \rho_k^T Q_k \rho_k)}{\operatorname{Tr}(H_k^2)} H_k, \tag{2.17}$$

$\|\cdot\|$ denotes the Euclidean norm of a vector.

To safeguard on the possibly of generating undefined Q_{k+1}, we propose to use our updating scheme for Q_{k+1}:

$$Q_{k+1} = \begin{cases} Q_k + \dfrac{(\rho_k^T \mu_k - \rho_k^T Q_k \rho_k)}{\operatorname{Tr}(H_k^2)} H_k; & \|\rho_k\| > 10^{-4}, \\ Q_k; & \text{otherwise.} \end{cases} \tag{2.18}$$

Now, we can describe the algorithm for our proposed method as follows:

Algorithm 2.1 (2-MFDN).

Step 1. Choose an initial guess x_0 and $Q_0 = I$, let $k := 0$.

Step 2 (Compute $F(x_k)$). If $\|F(x_k)\| \leq \epsilon_1$ stop, where $\epsilon_1 = 10^{-4}$.

Step 3. If $k := 0$ define $x_1 = x_0 - Q_0^{-1} F(x_0)$. Else if $k := 1$ set $\rho_k = s_k$ and $\mu_k = y_k$ and go to 5.

Step 4. If $k \geq 2$ compute ν_1, ν_0, and β via (2.4)–(2.6), respectively and find ρ_k and μ_k using (2.7) and (2.8), respectively. If $\rho_k^T \mu_k \leq 10^{-4} \|\rho_k\|_2 \|\mu_k\|_2$ set $\rho_k = s_k$ and $\mu_k = y_k$.

Step 5. Let $x_{k+1} = x_k - Q_k^{-1} F(x_k)$ and update Q_{k+1} as defined by (2.17).

Step 6. Check if $\|\rho_k\|_2 \geq \epsilon_1$, if yes retain Q_{k+1}, that is, computed by Step 5. Else set, $Q_{k+1} = Q_k$.

Step 7. Set $k := k + 1$ and go to 2.

3. Numerical Results

In this section, we analyze the performance of 2-MFDN method compared to four Newton-Like methods. The codes are written in MATLAB 7.0 with a double-precision computer, the stopping criterion used is

$$\|\rho_k\| + \|F(x_k)\| \leq 10^{-4}. \tag{3.1}$$

The methods that we considered are

(i) Newton's method (NM).

(ii) Chord (fixed) Newton-method (FN).

(iii) The Broyden method (BM).

(iv) MFDN stands for the method proposed by Leong et al. [6].

The identity matrix has been chosen as an initial approximate Jacobian. Six (2.4) different dimensions are performed on the benchmark problems, that is, 25, 50, 100, 500, 1000, and 250000.

The symbol "−" is used to indicate a failure due to:

(i) the number of iteration is at least 500 but no point of x_k that satisfies (3.1) is obtained;

(ii) CPU time in second reaches 500;

(iii) insufficient memory to initiate the run.

In the following, we illustrate some details on the benchmarks test problems.

Problem 1. Trigonometric system of Shin et al. [12]:

$$f_i(x) = \cos(x_i) - 1, \quad i = 1, 2, \ldots, n, \qquad x_0 = (0.87, 0.87, \ldots, 0.87). \tag{3.2}$$

Problem 2. Artificial function:

$$f_i(x) = \ln(x_i) \cos\left(\left(1 - \left(1 + \left(x^T x\right)^2\right)^{-1}\right)\right) \exp\left(\left(\left(1 - \left(1 + \left(x^T x\right)^2\right)^{-1}\right)\right)\right), \quad i = 1, 2, \ldots, n,$$
$$x_0 = (2.5, 2.5, \ldots, 2.5). \tag{3.3}$$

Problem 3. Artificial function:

$$f_1(x) = \cos x_1 - 9 + 3x_1 + 8 \exp x_2,$$
$$f_i(x) = \cos x_i - 9 + 3x_i + 8 \exp x_{i-1}, \quad i = 2, \ldots, n, \tag{3.4}$$
$$(5, 5, \ldots, 5).$$

Problem 4. Trigonometric function of Spedicato [13]:

$$f_i(x) = n - \sum_{j=1}^{n} \cos x_j + i(1 - \cos x_i) - \sin x_i, \quad i = 1, \ldots, n, \qquad x_0 = \left(\frac{1}{n}, \frac{1}{n}, \ldots, \frac{1}{n}\right). \tag{3.5}$$

Problem 5. Spares system of Shin et al. [12]:

$$f_i(x) = x_i x_{i+1} - 1, \qquad f_n(x) = x_n x_1 - 1, \quad i = 1, 2, \ldots, n - 1,$$
$$x_0 = (0.5, 0.5, \ldots, .5). \tag{3.6}$$

Table 1: Numerical results of NM, FN, BM, MFDN, and 2-MFDN methods.

Prob	Dim	NM		FN		BM		MFDN		2-MFDN	
		NI	CPU	NI	CPU	NI	CPU	NI	CPU	NI	CPU
1	25	7	0.062	236	0.047	12	0.005	22	0.008	25	0.006
	50	7	0.094	282	0.140	12	0.015	23	0.009	27	0.008
	100	7	0.872	356	0.168	12	0.018	24	0.016	27	0.013
	500	7	16.988	—	—	13	4.092	26	0.048	29	0.031
	1000	7	101.471	—	—	14	7.167	26	0.064	31	0.048
	250000	—	—	—	—	—	—	32	13.949	38	9.982
2	25	5	0.062	—	—	7	0.016	8	0.016	5	0.016
	50	6	0.109	—	—	7	0.031	8	0.019	5	0.018
	100	6	0.593	—	—	7	0.312	8	0.033	5	0.031
	500	6	16.1305	—	—	7	2.168	8	0.038	5	0.033
	1000	6	107.8747	—	—	7	8.736	8	0.068	5	0.047
	250000	—	—	—	—	—	—	9	4.919	6	3.666
3	25	—	—	—	—	—	—	24	0.031	14	0.018
	50	—	—	—	—	—	—	24	0.031	14	0.022
	100	—	—	—	—	—	—	24	0.047	14	0.031
	500	—	—	—	—	—	—	24	0.068	14	0.035
	1000	—	—	—	—	—	—	24	0.125	14	0.062
	250000	—	—	—	—	—	—	25	12.612	15	8.284
4	25	6	0.031	68	0.062	11	0.016	12	0.009	12	0.009
	50	7	0.187	114	0.125	12	0.032	13	0.028	13	0.028
	100	7	0.842	181	0.718	13	0.312	14	0.036	12	0.026
	500	7	18.985	415	26.801	14	6.099	15	0.047	11	0.032
	1000	7	131.743	—	—	19	28.205	15	0.062	14	0.039
	250000	—	—	—	—	—	—	15	9.001	31	16.115
5	25	5	0.015	—	—	5	0.003	6	0.002	4	0.001
	50	—	—	—	—	5	0.021	6	0.016	4	0.011
	100	—	—	—	—	5	0.025	6	0.019	4	0.015
	500	—	—	—	—	5	1.567	6	0.029	4	0.019
	1000	—	—	—	—	5	6.115	6	0.032	5	0.024
	250000	—	—	—	—	—	—	7	2.464	5	2.371
6	25	13	0.190	—	—	—	—	22	0.018	23	0.016
	50	13	0.328	—	—	—	—	23	0.038	24	0.031
	100	14	1.232	—	—	—	—	24	0.064	27	0.041
	500	15	39.578	—	—	—	—	27	0.187	30	0.109
	1000	17	280.830	—	—	—	—	29	0.216	31	0.160
	250000	—	—	—	—	—	—	38	39.104	45	30.451
7	25	5	0.032	21	0.016	5	0.015	6	0.005	4	0.005
	50	5	0.047	36	0.031	5	0.018	6	0.010	4	0.009
	100	5	0.390	62	0.203	5	0.047	6	0.019	4	0.015
	500	5	12.948	127	7.379	5	1.373	6	0.056	4	0.032
	1000	5	89.357	134	30.997	5	6.536	6	0.061	4	0.032
	250000	—	—	—	—	—	—	6	3.524	4	2.777

Problem 6. Artificial function:

$$f_i(x) = n(x_i - 3)^2 + \frac{\cos(x_i - 3)}{2} - \frac{x_i - 2}{\exp(x_i - 3) + \log\left(x_i^2 + 1\right)}, \quad i = 1, 2, \ldots, n,$$
$$x_0 = (-3, -3, -3, \ldots, -3). \tag{3.7}$$

Problem 7 (see [14]).

$$f_1(x) = x_1,$$
$$f_i(x) = \cos x_{i+1} + x_i - 1, \quad i = 2, 3, \ldots, n, \qquad x_0 = (0.5, 0.5, \ldots, .5). \tag{3.8}$$

From Table 1, it is noticeably that using the 2-step approach in building the diagonal updating scheme has significantly improved the performance of the one-step diagonal variant of Newton's method (MFDN). This observation is most significant when one considers CPU time and number of iterations, particularly as the systems dimensions increase. In addition, it is worth mentioning that, the result of 2-MFDN method in solving Problem 3 shows that the method could be a good solver even when the Jacobian is nearly singular.

The numerical results presented in this paper shows that 2-MFDN method is a good alternative to MFDN method especially for extremely large systems.

4. Conclusions

In this paper, a new variant of secant method for solving large-scale system of nonlinear equations has been developed (2-MFDN). The method employs a two-step, 3-point scheme to update the Jacobian approximation into a nonsingular diagonal matrix, unlike the single-step method. The anticipation behind this approach is enhanced by the Jacobian approximation. Our method requires very less computational cost and number of iterations as compared to the MFDN method of Leong et al. [6]. This is more noticeable as the dimension of the system increases. Therefore, from the numerical results presented, we can wind up that, our method (2-MFDN) is a superior algorithm compared to NM, FN, BM, and MFDN methods in handling large-scale systems of nonlinear equations.

References

[1] J. E. Dennis, Jr. and R. B. Schnabel, *Numerical Methods for Unconstrained Optimization and Nonlinear Equations*, Prentice Hall, Englewood Cliffs, NJ, USA, 1983.

[2] R. S. Dembo, S. C. Eisenstat, and T. Steihaug, "Inexact Newton methods," *SIAM Journal on Numerical Analysis*, vol. 19, no. 2, pp. 400–408, 1982.

[3] K. Natasa and L. Zorna, "Newton-like method with modification of the righ-thand vector," *Journal of Computational Mathematics*, vol. 71, pp. 237–250, 2001.

[4] M. Y. Waziri, W. J. Leong, M. A. Hassan, and M. Monsi, "A low memory solver for integral equations of chandrasekhar type in the radiative transfer problems," *Mathematical Problems in Engineering*, vol. 2011, Article ID 467017, 12 pages, 2011.

[5] B. Lam, "On the convergence of a quasi-Newton method for sparse nonlinear systems," *Mathematics of Computation*, vol. 32, no. 142, pp. 447–451, 1978.

[6] W. J. Leong, M. A. Hassan, and M. Waziri Yusuf, "A matrix-free quasi-Newton method for solving large-scale nonlinear systems," *Computers & Mathematics with Applications*, vol. 62, no. 5, pp. 2354–2363, 2011.

[7] J. E. Dennis, Jr. and H. Wolkowicz, "Sizing and least-change secant methods," *SIAM Journal on Numerical Analysis*, vol. 30, no. 5, pp. 1291–1314, 1993.

[8] J. A. Ford and I. A. Moghrabi, "Alternating multi-step quasi-Newton methods for unconstrained optimization," *Journal of Computational and Applied Mathematics*, vol. 82, no. 1-2, pp. 105–116, 1997, 7th ICCAM 96 Congress (Leuven).

[9] J. A. Ford and I. A. Moghrabi, "Multi-step quasi-Newton methods for optimization," *Journal of Computational and Applied Mathematics*, vol. 50, no. 1–3, pp. 305–323, 1994.

[10] M. Farid, W. J. Leong, and M. A. Hassan, "A new two-step gradient-type method for large-scale unconstrained optimization," *Computers & Mathematics with Applications*, vol. 59, no. 10, pp. 3301–3307, 2010.

[11] J. A. Ford and S. Thrmlikit, "New implicite updates in multi-step quasi-Newton methods for unconstrained optimization," *Journal of Computational and Applied Mathematics*, vol. 152, pp. 133–146, 2003.

[12] B.-C. Shin, M. T. Darvishi, and C.-H. Kim, "A comparison of the Newton-Krylov method with high order Newton-like methods to solve nonlinear systems," *Applied Mathematics and Computation*, vol. 217, no. 7, pp. 3190–3198, 2010.

[13] E. Spedicato, "Cumputational experience with quasi-Newton algorithms for minimization problems of moderatetly large size," Tech. Rep. CISE-N-175 3, pp. 10–41, 1975.

[14] A. Roose, V. L. M. Kulla, and T. Meressoo, *Test Examples of Systems of Nonlinear Equations.*, Estonian Software and Computer Service Company, Tallin, Estonia, 1990.