TÜBİTAK

Turkish Journal of Mathematics

http://journals.tubitak.gov.tr/math/

Research Article

Turk J Math
(2015) 39: 335 – 345

doi:10.3906/mat-1404-41

On Broyden-like update via some quadratures for solving nonlinear systems of equations

Hassan MOHAMMAD*, Muhammad Yusuf WAZIRI
Department of Mathematical Sciences, Faculty of Sciences, Bayero University, Kano, Kano State, Nigeria

Received: 14.04.2014 • **Accepted/Published Online:** 14.01.2015 • **Printed:** 29.05.2015

Abstract: In this work, we propose a new alternative approximation based on the quasi-Newton approach for solving systems of nonlinear equations using the average of midpoint and Simpson's quadrature. Our goal is to enhance the efficiency of the method (Broyden's method) by reducing the number of iterations it takes to reach a solution. Local convergence analysis and computational results showing the relative efficiency of the proposed method are given.

Key words: Broyden's method, superlinear convergence, quadrature formulae, predictor-corrector, nonlinear systems

1. Introduction

Consider the numerical solution of the systems of nonlinear equations of the form

$$F(\mathrm{x}) = 0. \tag{1}$$

One of the methods used to solve (1) is Newton's method. It is famous with quadratic order of convergence under some mild assumptions [13]. Despite its good convergence property, Newton's method has some shortcomings, such as computing and storing Jacobian matrices, solving systems of linear equation in every iteration, and inefficiency in handling large-scale systems. In an attempt to reduce the computational cost of Newton's method, quasi-Newton methods have been introduced [2]. These methods approximate the Jacobian matrix or its inverse using a derivative-free matrix that is updated in each iteration, and its order of convergence was proven to be superlinear [8]. The most successful and simplest quasi-Newton method for solving nonlinear systems of equations is the Broyden method. Broyden's method is given by

$$\mathrm{x}_{k+1} = \mathrm{x}_k - B_k^{-1}F(\mathrm{x}_k), \quad k = 0, 1, 2, ... \tag{2}$$

where matrix B_k is the approximation of $F'(\mathrm{x}_k)$, such that the quasi-Newton equation

$$B_{k+1}(\mathrm{x}_{k+1} - \mathrm{x}_k) = F(\mathrm{x}_{k+1}) - F(\mathrm{x}_k) \tag{3}$$

is satisfied for each k.

It is vital to report that Broyden's method needs n^2 (n is the length of the vector x) storage location; therefore, for large-scale systems, this might lead to severe memory constraints. From the early 1970s, there has

*Correspondence: hmuhammad.mth@buk.edu.ng
2010 *AMS Mathematics Subject Classification:* 65K05, 90C53, 41A25, 65D32, 34G20.

been serious attention paid to the issue of reducing the amount of storage required for the iterative methods. Several modifications of Broyden's method were given in order to reduce its computational cost; see, for example, [1, 3, 5, 9, 15, 16, 18, 24] and references therein. The Broyden rank reduction method involves approaching the update matrix by a low-rank matrix. These methods are sometimes called "limited memory Broyden methods" [19, 24]. In these methods, the amount needed for storing the Broyden matrix is reduced from n^2 to $2pn$ storage locations where p is a predefined and fixed positive integer with $1 \leq p \leq n$.

The emergence of quasi-Newton's methods has contributed to the improvement of solving nonlinear systems by reducing the cost of computing and storing the Jacobian in Newton's method, but the number of iterations needed to converge to a solution has increased, which inversely reduced the convergence order from quadratic to superlinear. In [14], Mamat et al. proposed a Broyden-like method using the trapezoidal rule to solve a system of nonlinear equations, and numerical testing in that paper showed that the new method converges with fewer iterations than Broyden's method. Motivated by this idea, we employ the average of the midpoint and Simpson's rule, which exhibits a higher approximation order than that of trapezoidal rule.

Our aim in this work is to present an alternative method that will reduce the number of iterations required by the classical Broyden method to converge to a solution preserving its local order of convergence. This is achieved by improving the efficiency of the method using the average of the midpoint and Simpson's quadratures to approximate the integral from the analog of the fundamental theorem of calculus in $\mathbb{R}^n$ [6], and then replacing the Jacobian obtained with Broyden's matrix. The proposed methods are two-step in nature, where the first step is the classical Broyden method and the second step is a quadrature-based Broyden method.

The rest of this paper is organized as follows. Section 2 is designed for the derivation of the new method, while the local convergence analysis of the proposed method is given in section 3. In section 4 the computational experiment of the proposed method is given and compared to the existing classical methods, and finally the conclusion comes in section 5.

2. New alternative approximations

In this section, we present our new scheme. Consider the following result, whose proof can be found in [17].

Lemma 2.1 *[17] Let $F : C \subset \mathbb{R}^n \to \mathbb{R}^n$ be continuously differentiable on a convex set Ω. Then for any u, $v \in \Omega$,*

$$F(v) - F(u) = \int_0^1 F'(u + \eta(v-u))(v-u)d\eta. \tag{4}$$

Let x be in the neighborhood of a solution $\mathrm{x}^* \in C$. Then, from (4),

$$F(\mathrm{x}^*) = F(\mathrm{x}) + \int_0^1 F'(\mathrm{x} + \eta(\mathrm{x}^* - \mathrm{x}))(\mathrm{x}^* - \mathrm{x})d\eta,$$

and letting x be the iterate x_k we have

$$F(\mathrm{x}^*) = F(\mathrm{x}_k) + \int_0^1 F'(\mathrm{x}_k + \eta(\mathrm{x}^* - \mathrm{x}_k))(\mathrm{x}^* - \mathrm{x}_k)d\eta. \tag{5}$$

Approximating the integral in (5) in the interval $[0, 1]$ when $\eta = 0$ by the average of the midpoint and Simpson's quadrature rules yields

$$F(\mathrm{x}^*) \approx F(\mathrm{x}_k) + \frac{1}{12}[F'(\mathrm{x}_k) + 10F'(\frac{\mathrm{x}^* + \mathrm{x}_k}{2}) + F'(\mathrm{x}^*)](\mathrm{x}^* - \mathrm{x}_k).$$

Since x^* is a solution of $F(\mathrm{x}) = 0$, it follows that

$$-F(\mathrm{x}_k) = \frac{1}{12}[F'(\mathrm{x}_k) + 10F'(\frac{\mathrm{x}^* + \mathrm{x}_k}{2}) + F'(\mathrm{x}^*)](\mathrm{x}^* - \mathrm{x}_k).$$

Letting x_{k+1} be the next approximation to x^*, we have

$$\mathrm{x}_{k+1} = \mathrm{x}_k - 12[F'(\mathrm{x}_k) + 10F'(\frac{\mathrm{x}_{k+1} + \mathrm{x}_k}{2}) + F'(\mathrm{x}_{k+1})]^{-1}F(\mathrm{x}_k), \tag{6}$$

which is an implicit equation.

To obtain the explicit form, we adopt the technique used in [6, 7, 11]. We compute the $(k+1)th$ iterate on the right hand side of (6) by replacing $F'(\frac{\mathrm{x}_{k+1}+\mathrm{x}_k}{2})$ with $F'(\frac{\mathrm{z}_k+\mathrm{x}_k}{2})$, where $\mathrm{z}_k = \mathrm{x}_k - F'(\mathrm{x}_k)^{-1}F(\mathrm{x}_k)$ is the Newton iterate, which is the predictor. Thus,

$$\mathrm{x}_{k+1} = \mathrm{x}_k - 12[F'(\mathrm{x}_k) + 10F'(\mathrm{w}_k) + F'(\mathrm{z}_k)]^{-1}F(\mathrm{x}_k) \quad k = 0, 1, \ldots \tag{7}$$

is the corrector, which is the method derived by Hafiz and Baghat [11] where, $\mathrm{w}_k = \frac{\mathrm{z}_k+\mathrm{x}_k}{2} = \mathrm{x}_k - \frac{1}{2}F'(\mathrm{x}_k)^{-1}F(\mathrm{x}_k)$.

Here, we choose to approximate $F'(\mathrm{x}_k)$, $F'(\mathrm{w}_k)$ and $F'(\mathrm{z}_k)$ using Broyden's update matrices; that is, $B(\mathrm{x}_k)$, $B(\mathrm{w}_k)$, and $B(\mathrm{z}_k)$, respectively. By letting

$$B_k = B(\mathrm{x}_k) + 10B(\mathrm{w}_k) + B(\mathrm{z}_k), \tag{8}$$

we obtain the two-step iterative scheme of the *midpoint-Simpson Broyden method*:

$$\begin{aligned} z_k &= \mathrm{x}_k - B(\mathrm{x}_k)^{-1}F(\mathrm{x}_k), \\ \mathrm{x}_{k+1} &= \mathrm{x}_k - 12B_k^{-1}F(\mathrm{x}_k), \quad \text{for} \quad k = 0, 1, ... \end{aligned} \tag{9}$$

Using simple algebra one can easily verify that the midpoint-Simpson Broyden method as member of the quasi-Newton family satisfies (3).

3. Convergence analysis

In this section the local order of convergence of our proposed method is proven to be superlinear. The following results will be useful in proving the main theorem of this section.

Definition 3.1 (q-Superlinearly convergence) *[13] Let $\{x_k\} \subset \mathbb{R}^n$ and $x^* \in \mathbb{R}^n$. Then $x_k \to x^*$* q-superlinearly *if*

$$\lim_{k\to\infty} \frac{\|x_{k+1} - x^*\|}{\|x_k - x^*\|} = 0.$$

Lemma 3.2 *[20] Let $F : \mathbb{R}^n \to \mathbb{R}^n$ be continuously differentiable on an open convex set $\Omega \subset \mathbb{R}^n$, $x \in \Omega$. If $F'(x)$ is Lipschitz continuous with Lipschitz constant λ, then for any $u, v \in \Omega$*

$$\|F(v) - F(u) - F'(x)(v - u)\| \leq \lambda \max\{\|v - x\|, \|u - x\|\}\|v - u\|.$$

Moreover, if $F'(x)$ is invertible, then there exist ϵ and $\rho > 0$ such that

$$\frac{1}{\rho}\|v - u\| \leq \|F(v) - F(u)\| \leq \rho\|v - u\|,$$

for all $u, v \in \Omega$ for which $\max\{\|u - x\|, \|v - x\|\} \leq \epsilon$.

Lemma 3.3 *[20] Let $x_k \in \mathbb{R}^n$, $k \geq 0$. If $\{x_k\}$ converges q-superlinearly to $x^* \in \mathbb{R}^n$, then*

$$\lim_{k\to\infty} \frac{\|x_{k+1} - x_k\|}{\|x_k - x^*\|} = 1.$$

Hence, we present the main result.

Theorem 3.4 *Let $F : \mathbb{R}^n \to \mathbb{R}^n$ satisfy the hypothesis of Lemma 3.2 on the set Ω. Let $\{B_k\}$ be a sequence of nonsingular matrices in $\mathcal{L}(\mathbb{R}^n)$- the space of real matrices of order n. Suppose for some x_0 the sequence $\{x_k\}$ generated by (9) remains in Ω and $\lim_{k\to\infty} x_k = x^*$ where for each k, $x_k \neq x^*$. Then $\{x_k\}$ converges q-superlinearly to x^* and $F(x^*) = 0$ if and only if*

$$\lim_{k\to\infty} \frac{\|(\frac{1}{12}B_k - F'(x^*))s_k\|}{\|s_k\|} = 0, \tag{10}$$

where $s_k = x_{k+1} - x_k$ and B_k is given in (8).

Proof Suppose Eq. (10) holds; then Eq. (9) gives

$$\begin{aligned} 0 &= \frac{1}{12}B_k s_k + F(x_k) \\ &= \frac{1}{12}B_k s_k + F(x_k) - F'(x^*)s_k + F'(x^*)s_k. \end{aligned}$$

Thus,

$$-F(x_{k+1}) = (\frac{1}{12}B_k - F'(x^*))s_k + (-F(x_{k+1})) + F(x_k) + F'(x^*)s_k. \tag{11}$$

Then, using vector norm properties and Lemma 3.2, we have

$$\begin{aligned} \frac{\|F(x_{k+1})\|}{\|s_k\|} &\leq \frac{\|(\frac{1}{12}B_k - F'(x^*))s_k\|}{\|s_k\|} + \frac{\|(-F(x_{k+1})) + F(x_k) + F'(x^*)s_k\|}{\|s_k\|} \\ &\leq \frac{\|(\frac{1}{12}B_k - F'(x^*))s_k\|}{\|s_k\|} + \lambda \max\{\|x_{k+1} - x^*\|, \|x_k - x^*\|\}. \end{aligned}$$

From (10) and that $x_k \to x^*$ for all k,

$$\begin{aligned} \lim_{k\to\infty} \frac{\|F(x_{k+1})\|}{\|s_k\|} &\leq \lim_{k\to\infty} \frac{\|(\frac{1}{12}B_k - F'(x^*))s_k\|}{\|s_k\|} + \lambda \lim_{k\to\infty} \max\{\|x_{k+1} - x^*\|, \|x_k - x^*\|\} \\ &= 0. \end{aligned} \tag{12}$$

Thus,

$$\begin{aligned} F(x^*) &= F(\lim_{k\to\infty} x_k) \\ &= \lim_{k\to\infty} F(x_k) \\ &= 0. \end{aligned}$$

Since $F'(x^*)$ is nonsingular, by Lemma 3.2 there exist $\rho > 0, k_0 \geq 0$, such that

$$\|F(x_{k+1})\| = \|F(x_{k+1}) - F(x^*)\| \geq \frac{1}{\rho}\|x_{k+1} - x^*\|. \tag{13}$$

Now, for all $k \geq k_0$, (12) and (13) give

$$\begin{aligned} 0 &= \lim_{k\to\infty} \frac{\|F(x_{k+1})\|}{\|s_k\|} \\ &\geq \lim_{k\to\infty} \frac{1}{\rho}\frac{\|x_{k+1} - x^*\|}{\|s_k\|} \\ &\geq \lim_{k\to\infty} \frac{1}{\rho}\frac{\|x_{k+1} - x^*\|}{\|x_k - x^*\| + \|x_{k+1} - x^*\|} \\ &= \lim_{k\to\infty} \frac{\frac{1}{\rho}t_k}{1 + t_k}, \quad \text{where } t_k = \frac{\|x_{k+1} - x^*\|}{\|x_k - x^*\|}, \end{aligned}$$

which implies that $\lim_{k\to\infty} t_k = 0$. Hence, $\{x_k\}$ converges q-superlinearly to x^*.

Conversely, suppose that $\{x_k\}$ converges q-superlinearly to x^* and $F(x^*) = 0$. By Lemma 3.2, $\exists \rho > 0$ such that

$$\|F(x_{k+1})\| \leq \rho\|x_{k+1} - x^*\|.$$

Now,

$$\begin{aligned} 0 = \lim_{k\to\infty} \frac{\|x_{k+1} - x^*\|}{\|x_k - x^*\|} &\geq \lim_{k\to\infty} \frac{\|F(x_{k+1})\|}{\rho\|x_k - x^*\|} \\ = \lim_{k\to\infty} \frac{\|F(x_{k+1})\|}{\rho\|s_k\|}\frac{\|s_k\|}{\|x_k - x^*\|}. \end{aligned}$$

Then, using Lemma 3.3, we obtain

$$\lim_{k\to\infty} \frac{\|F(x_{k+1})\|}{\|s_k\|} = 0. \tag{14}$$

From Eq. (11), we have

$$\begin{aligned} \lim_{k\to\infty} \frac{\|(\frac{1}{12}B_k - F'(x^*))s_k\|}{\|s_k\|} &\leq \lim_{k\to\infty} \frac{\|F(x_{k+1})\|}{\|s_k\|} + \lim_{k\to\infty} \frac{\| - F(x_{k+1}) + F(x_k) + F'(x^*)s_k\|}{\|s_k\|} \\ &\leq 0 + \lambda \lim_{k\to\infty} \max\{\|x_k - x^*\|, \|x_{k+1} - x^*\|\}. \end{aligned}$$

Since $\{x_k\}$ converges to x^*, $\lim_{k\to\infty} \|x_k - x^*\| = 0$. This proves that

$$\lim_{k\to\infty} \frac{\|(\frac{1}{12}B_k - F'(x^*))s_k\|}{\|s_k\|} = 0.$$

□

4. Numerical results

In this section, we compare the performance of our proposed method with that of the classical Broyden method (CB), modified autoadaptative limited memory Broyden method (LMB) [24], and trapezoidal Broyden method (TB) [14]. We denote by MSB the method defined by (9). An identity matrix is used as an initial approximation to the Jacobian matrix in the CB, LMB, TB, and MSB methods. We use ten test functions in order to check the effectiveness of the proposed methods. For the last six of them, we consider seven instances of dimension n, namely $n = 5, 15, 65, 165, 365, 665, 1065$. This makes a total of forty-six problems. x_0 stands for initial approximation to the solution in all the tested problems. In Tables 1 and 2 we present results on the following information: the number of iterations (Iter) needed to converge to a solution and the CPU time (in seconds).

A failure is reported (denoted by '-') if any of the following situations occur during the iteration process: the number of iterations and/or the CPU time (in seconds) reaches 300, but no x_k satisfying $\|s_k\| + \|F(\mathrm{x}_k)\| \leq 10^{-8}$ is obtained; failure on execution of the code due to insufficient memory; and failure due to the approach to singular matrix during the iteration process.

We implement the four methods (CB, LMB, TB, and MSB) using MATLAB R2010a and the *tic- toc* command is used for reporting the CPU time. All computations were carried out on a PC with Intel COREi3 processor with 4 GB of RAM and CPU 2.30 GHz.

We compare the performance among the tested methods based on the performance profile presented by Dolan and Morе́ [10]. For n_s solvers and n_p problems, the performance profile $P : \mathbb{R} \to [0, 1]$ is defined as follows: let $\mathcal{P}$ and $\mathcal{S}$ be the set of problems and set of solvers respectively. For each problem $p \in \mathcal{P}$ and for each solver $s \in \mathcal{S}$ we define $t_{p,s} :=$ (number of iterations required to solve problem p by solver s). The performance ratio is given by $r_{p,s} := \frac{t_{p,s}}{\min\{t_{p,s} | s \in \mathcal{S}\}}$. Then the performance profile is defined by $P(\tau) := \frac{1}{n_p}\text{size}\{p \in \mathcal{P} | r_{p,s} \leq \tau\}, \forall \tau \in \mathbb{R}$, where $P(\tau)$ is the probability for solver $s \in \mathcal{S}$ that a performance ratio $r_{p,s}$ is within a factor $\tau \in \mathbb{R}$ of the best possible ratio.

List of the tested problems:

Problem 1 [12]

$F(x_1, x_2) = ((x_1 - 1)^2(x_1 - x_2), (x_2 - 2)^5 cos(\frac{2x_1}{x_2}))$.

Problem 2 [23]

$F(x_1, x_2) = (|x_1| + (x_2 - 1)^2 - 1, (x_1 - 1)^2 + |x_2| - 1)$.

Problem 3 [22]

$F(x_1, x_2, x_3) = (cosx_1 - 9 + 3x_1 + 8exp(x_2), cosx_2 - 9 + 3x_2 + 8exp(x_1), cosx_3 - x_3 - 1)$.

Problem 4 [1]

$F_i(x) = x_i - \frac{\sum x_j^3 + 1}{8} \qquad i = 1, \ldots, 4.$

Problem 5 [4]

$F_i(\mathrm{x}) = \mathrm{x}_i\mathrm{x}_{i+1} - 1,$

$F_n(\mathrm{x}) = \mathrm{x}_n\mathrm{x}_1 - 1$

$i = 1, 2, \ldots, n-1$ and $\mathrm{x}_0 = (0.5, 0.5, \ldots, 0.5)^T$.

Problem 6 (Artificial problem)

$F_i(\mathrm{x}) = (cos(\mathrm{x}_i) - 1)^2 - 1,$

$i = 1, 2, \ldots, n$ and $\mathrm{x}_0 = (1, 1, \ldots, 1)^T$.

Problem 7 [11]

$$F_i(\mathrm{x}) = \mathrm{x}_i^2 - \cos(\mathrm{x}_i - 1), \quad i = 1, 2, \ldots, n \text{ and } \mathrm{x}_0 = (1.5, 1.5, \ldots, 1.5)^T.$$

Problem 8 (Artificial problem)

$$F_1(\mathrm{x}) = -2\mathrm{x}_1^2 + 3\mathrm{x}_1 - 2\mathrm{x}_2 + 1,$$

$$F_i(\mathrm{x}) = -2\mathrm{x}_i^2 + 3\mathrm{x}_i - 2\mathrm{x}_{i-1} + 1,$$

$$F_n(\mathrm{x}) = -2\mathrm{x}_n^2 + 3\mathrm{x}_n - 2\mathrm{x}_{n-1} + 1$$

$i = 2, 3, \ldots, n-1$ and $\mathrm{x}_0 = (2, 2, \ldots, 2)$.

Problem 9 [21]

$$F_i(\mathrm{x}) = In(\mathrm{x}_i)cos(1 - (1 + x^T x)^2)^{-1})exp((1 - (1 + x^T x)^2)^{-1})$$

$i = 1, 2, \ldots, n$ and $\mathrm{x}_0 = (2.5, 2.5, \ldots, 2.5)^T$.

Problem 10 [2]

$$F_1(\mathrm{x}) = (3 - ux_1)\mathrm{x}_1 - 2\mathrm{x}_2 + 1,$$

$$F_i(\mathrm{x}) = (3 - ux_i)\mathrm{x}_i - \mathrm{x}_{i-1} - 2\mathrm{x}_{i+1} + 1,$$

$$F_n(\mathrm{x}) = (3 - ux_n)\mathrm{x}_n - \mathrm{x}_{n-1} + 1.$$

$i = 2, 3, ..., n-1$, $u = 2$ and $\mathrm{x}_0 = (0, 0, \ldots, 0)^T$.

The nuerical results in Tables 1 and 2 clearly show that our method has reduced the number of iterations required to solve the tested problems. In Figures 1 and 2, MSB performs best with probability of about 0.60 and 0.85, respectively. Although CB manages to solve all the problems in Figure 1, MSB solves about 60% of the problems with a lower number of iterations. We can observe from Figure 2 that for the fraction of $\tau > 4$, MSB and LMB are the best solvers, but for $\tau < 4$, MSB outperforms the rest of the solvers. In short, MSB solves and wins 91.3%, TB solves and wins 23.9%, and CB solves and wins 6.52% of the tested problems.

Table 1. Numerical results for problems 1–4.

Problem	Guess	CB		LMB		TB		MSB	
		Iter	CPU	Iter	CPU	Iter	CPU	Iter	CPU
1	(1,1)	12	0.3995	14	0.2711	8	0.3927	7	0.4604
	(1.7,1.5)	36	1.0598	-	-	-	-	-	-
	(1.9,2)	5	0.2067	7	0.1365	4	0.2317	4	0.2989
2	(0.5,0.5)	8	0.8697	14	0.2946	6	0.3210	5	0.3540
	(-0.5,-0.5)	7	0.2808	8	0.1994	7	0.3587	4	0.2654
	(-1,-1)	8	0.3057	13	0.2606	1	0.1302	5	0.3518
3	(1,2,-2)	24	0.8310	-	-	-	-	-	-
	(1.5,2.3,-1.8)	10	0.3871	-	-	-	-	-	-
	(2,1,-1)	7	0.2882	-	-	-	-	-	-
4	(0.5,0.5,0.5,0.5)	5	0.4334	7	0.3534	4	0.5025	4	0.6306
	(1.5,1.5,1.5,1.5)	7	0.5770	7	0.3311	6	0.7274	4	0.6267
	(-3,-3,-3,-3)	11	0.8585	10	0.4356	8	0.9042	7	1.0145

Table 2. Numerical results for problems 5–10.

Problem	DIM	CB		LMB		TB		MSB	
		Iter	CPU	Iter	CPU	Iter	CPU	Iter	CPU
5	5	6	0.0223	12	0.0127	5	0.0343	4	0.0048
	15	6	0.0039	12	0.0033	5	0.0026	4	0.0038
	65	6	0.0125	12	0.0368	5	0.0058	4	0.0149
	165	6	0.0155	13	0.1424	5	0.0509	4	0.0438
	365	6	0.1485	13	1.0261	5	0.1594	4	0.1158
	665	6	0.2528	13	6.0735	5	0.4003	4	0.3878
	1065	7	0.8423	13	27.2486	5	1.3092	4	1.1270
6	5	6	0.0159	8	0.0043	5	0.0245	4	0.0044
	15	6	0.0020	8	0.0037	5	0.0045	4	0.0048
	65	6	0.0070	8	0.0585	5	0.0073	4	0.0167
	165	6	0.0558	8	0.0919	5	0.0748	4	0.0607
	365	6	0.1250	8	0.6257	5	0.1481	4	0.1441
	665	7	0.2767	8	3.4729	5	0.4011	5	0.4623
	1065	7	0.7132	8	15.9695	5	1.3252	5	1.4734
7	5	11	0.0027	8	0.0025	7	0.0052	4	0.0041
	15	11	0.0059	8	0.0036	7	0.0063	4	0.0051
	65	11	0.0131	8	0.0382	7	0.0171	5	0.0111
	165	11	0.0488	8	0.0926	7	0.0387	5	0.0536
	365	11	0.1291	8	0.6695	7	0.1603	5	0.1435
	665	11	0.4487	8	3.5873	7	0.6038	5	0.4900
	1065	11	1.3439	8	15.8269	7	1.7910	5	1.5636
8	5	9	0.0057	14	0.0070	6	0.0092	6	0.0063
	15	9	0.0051	14	0.0081	6	0.0054	6	0.0066
	65	-	-	14	0.0292	6	0.0188	6	0.0223
	165	-	-	14	0.1689	6	0.0633	6	0.0518
	365	-	-	14	1.1364	6	0.1562	6	0.1690
	665	-	-	14	6.4593	6	0.5150	6	0.6352
	1065	-	-	14	29.4250	13	3.5660	6	1.6617
9	5	7	0.0024	10	0.5386	6	0.0180	5	0.0275
	15	7	0.0044	11	0.0089	6	0.0048	5	0.0037
	65	8	0.0101	13	0.1465	6	0.0297	5	0.0149
	165	8	0.0594	13	0.5781	6	0.0738	5	0.0717
	365	8	0.1221	13	2.8815	6	0.1402	5	0.1625
	665	8	0.3141	13	16.8575	6	0.5035	5	0.4776
	1065	8	0.9212	13	66.8782	6	1.5516	5	1.4875
10	5	19	0.0040	-	-	-	-	-	-
	15	39	0.0125	-	-	-	-	-	-
	65	109	0.0928	-	-	-	-	-	-
	165	228	0.4314	-	-	-	-	-	-
	365	-	-	-	-	-	-	-	-
	665	-	-	-	-	-	-	-	-
	1065	-	-	-	-	-	-	-	-

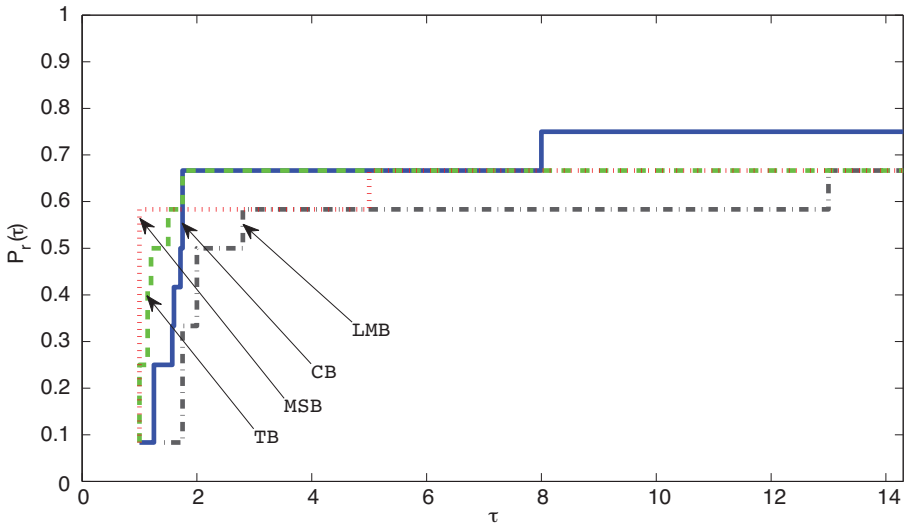

Figure 1. Performance profile of CB, LMB, TB, and MSB methods with respect to number of iterations for problems 1–4.

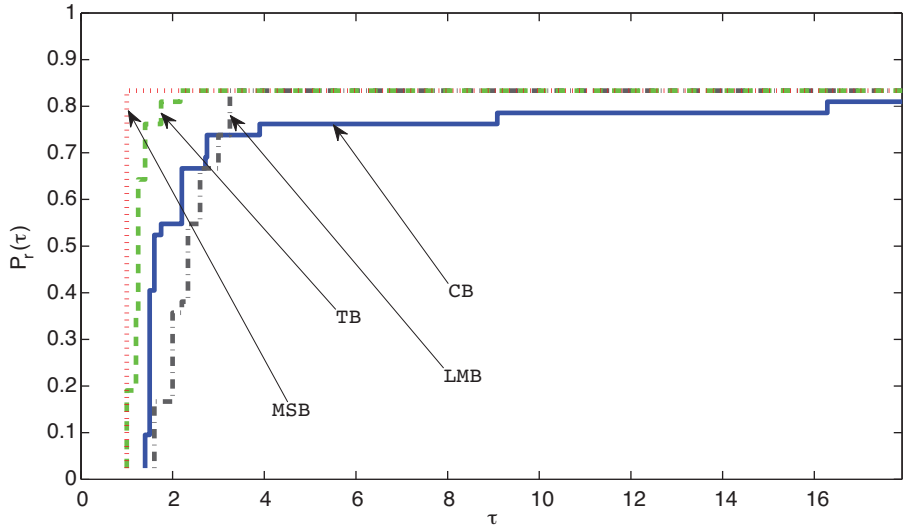

Figure 2. Performance profile of CB, LMB, TB, and MSB methods with respect to number of iterations for problems 510.

5. Conclusions

We have shown that one can use the approach of approximating Jacobian matrices with Broyden's matrices via a quadrature formula and we obtained an efficient update maintaining the local order of convergence of Broyden's method.

Numerical experiments show a strong indication that the newly proposed Broyden-like method exhibits enhanced performance (with respect to number of iterations) in most of the tested problems by comparison with the other variants. Finally, we can conclude that the use of numerical quadratures to approximate the Newton step is capable of improving the efficiency of the quasi-Newton (Broyden) method to an acceptable level.

Acknowledgment

The authors are grateful to the anonymous referees for their useful suggestions.

References

[1] Bierlaire M, Crittin F, Themans M. A multi-iterative method to solve systems of nonlinear equations. Eur J Oper Res 2007; 183: 20–41.

[2] Broyden CG. A class of methods for solving nonlinear simultaneous equations. Math Comput 1965; 19: 577–593.

[3] Buckley A, LeNir A. QN-like variable storage conjugate gradients. Math Program 1983; 27: 103–119.

[4] Byeong CS, Darvishi MT, Chang HK. A comparison of the Newton-Krylov method with high order Newton-like methods to solve nonlinear systems. Appl Math Comput 2010; 217: 3190–3198.

[5] Byrd RH, Nocedal J, Schnabel RB. Representations of Quasi-Newton Matrices and Their Use in Limited Memory Methods. Technical Report NAM-03. Evanston, IL, USA: Northwestern University, 1996.

[6] Cordero A, Torregrosa JR. Variants of Newton's method for functions of several variables. Appl Math Comput 2006; 183: 199–208.

[7] Cordero A, Torregrosa Juan R, Vassileva MP. Pseudocomposition: a technique to design predictor-corrector methods for systems of nonlinear equations. Appl Math Comput 2012; 218: 11496–11504.

[8] Dennis JE, Moré JJ. A characterization of superlinear convergence and its application to quasi-Newton methods. Math Comput 1974; 28: 549–560.

[9] Dhamacharoen A. An efficient hybrid method for solving systems of nonlinear equations. J Comput Appl Math 2014; 263: 59–68.

[10] Dolan ED, Moré JJ. Benchmarking optimization software with performance profiles. Math Program Ser A 2002; 91: 201–213.

[11] Hafiz MA, Bahgat SM. An efficient two-step iterative method for solving system of nonlinear equations. J Math Res 2012; 4: 28–34.

[12] Hueso JL, Martinéz E, Torregrosa JR. Modified Newton's method for systems of nonlinear equations with singular Jacobian. J Comput Appl Math 2009; 224: 77–83.

[13] Kelly CT. Iterative Methods for Linear and Nonlinear Equations. Philadelphia, PA, USA: SIAM, 1995.

[14] Mamat M, Muhammad K, Waziri MY. Trapezoidal Broyden's method for solving systems of nonlinear equations. Appl Math Sc 2014; 8: 251–260.

[15] Martinéz JM. A quasi-Newton method with modification of one column per iteration. Computing 1984; 33: 353-362.

[16] Martinéz JM, Zambaldi MC. An inverse column-updating method for solving large-scale nonlinear systems of equations. Optim Method Softw 1992; 1: 129–140.

[17] Ortega JM, Rheinboldt WC. Iterative Solution of Nonlinear Equations in Several Variables. New York, NY, USA: Academic Press, Inc., 1970.

[18] Schlenkrich S, Walther A. Global convergence of quasi-Newton methods based on adjoint Broyden updates. Appl Numer Math 2009; 59: 1120–1136.

[19] Van De Rotten B. A limited memory Broyden method to solve high-dimensional systems of nonlinear equations. PhD, University of Leiden, Leiden, the Netherlands, 2003.

[20] Van De Rotten B, Lunel SV. A Limited Memory Broyden Method to Solve High-Dimensional Systems of Nonlinear Equations. Technical Report MI 2003-2006. Leiden, the Netherlands: University of Leiden, 2003.

[21] Waziri MY, Leong WJ, Hassan MA. Diagonal Broyden-like method for large-scale systems of nonlinear equations. Malays J Math Sc 2012; 6: 59–73.

[22] Waziri MY, Leong WJ, Hassan MA. Jacobian-free diagonal Newton's method for solving nonlinear systems with singular Jacobian. Malays J Math Sc 2011; 5: 241–255.

[23] Xu H, Chang XW. Approximate Newton methods for nonsmooth equations. J Optimiz Theory App 1997; 93: 373–394.

[24] Ziani M, Guyomarch F. An autoadaptative limited memory Broyden's method to solve systems of nonlinear equations. Appl Math Comput 2008; 205: 202-211.