Journal of the
Nigerian Mathematical Society

Vol. 35, pp. 71-83, 2017

BARZILAI-BORWEIN-LIKE METHOD FOR SOLVING LARGE-SCALE NON-LINEAR SYSTEMS OF EQUATIONS

HASSAN MOHAMMAD

ABSTRACT. In this paper, a derivative-free Barzilai-Borwein-like algorithm is developed for solving large-scale non-linear systems of equations. The algorithm is based on approximating the Jacobian matrix in quasi-Newton manner using a scalar multiple of an identity matrix. Under suitable conditions, we show that the proposed algorithm is locally superlinearly convergent. Numerical results show that the proposed method is efficient for large-scale problems (up to 10^6) variables.

Keywords and phrases: Non-linear equations, Large-scale problems, Barzilai and Borwein method, Superlinear convergence
2010 Mathematical Subject Classification: 65K05 , 90C06, 90C52, 90C56, 49M30

1. INTRODUCTION

We present an iterative method for solving system of non-linear equations of the form

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

where $F : \mathbb{R}^n \to \mathbb{R}^n$ is continuously differentiable function, $F = (f_1, f_2, ..., f_n)^T$, $f_i : \mathbb{R}^n \to \mathbb{R}(i = 1, 2, ..., n)$. Newton's method is the most popular method used to solve (1), it converges locally with a quadratic rate of convergence [1]. The main drawback of Newton's method for large-scale problems is the need of computing and storing Jacobian matrix and solving system of linear equations in every iteration. As a remedy of these drawbacks, quasi-Newton methods have been introduced [2]. These methods are derivative-free, and enjoys superlinear rate of convergence [3]. For excellent review of quasi-Newton methods see [4, 5]. A suitable quasi-Newton method for solving (1) is Broyden method, it is given by

$$x_{k+1} = x_k - B_k^{-1}F(x_k), \quad k = 0, 1, 2, ... \tag{2}$$

Received by the editors February 10, 2016; Revised: June 08, 2016; Accepted: January 18, 2017
www.nigerianmathematicalsociety.org

where matrix B_k is the approximation of the Jacobian matrix in Newton's method, such that the quasi-Newton equation

$$B_{k+1}(x_{k+1} - x_k) = F(x_{k+1}) - F(x_k) \tag{3}$$

is satisfied for each k. Recently, some modifications of the Broyden update have been presented (see, for example,[6, 7] and reference therein). Although, qausi-Newton methods have increased the efficiency of Newton's method, still the requirement of storing a matrix is needed for these methods, which in turn makes them unsuitable for large-scale problems.

Waziri et al. [8] proposed an alternative approximation for the Newton step via diagonal updating (DBLM). The main anticipation behind their approach is to reduce the computational cost of computing Jacobian matrices when solving non-linear system of equations using Newton's method. The convergence of the proposed DBLM algorithm was proven to be linear and numerical experiments given in the paper shows the efficiency of the method compared to the existing ones like classical Newton's method, Broyden method and fixed Newton method. They consider

$$x_{k+1} = x_k - Q_k F(x_k), \tag{4}$$

where matrix Q_k is a diagonal matrix approximating the inverse Jacobian matrix. And $Q_{k+1} = Q_k + U_k$, U_k is also a diagonal matrix acting as a corrector such that the weak quasi Newton equation

$$v_k^T(Q_{k+1})v_k = v_k^T(x_{k+1} - x_k) \tag{5}$$

is satisfied for each k, where $v_k = F(x_{k+1}) - F(x_k)$.

Barzilai and Borwein (BB) [9] presented a two-point step size gradient methods for problem of minimizing strictly convex two dimensional quadratic function

$$\min f(x) = \frac{1}{2}x^T Ax - b^T x,$$

where $A \in \mathbb{R}^{2\times 2}$ is a real symmetric positive definite matrix and $b \in \mathbb{R}^2$ is constant. A remark was given in the paper that the presented algorithms are applicable to the solution of (1). Raydan [10], established a convergence result of the BB method for minimizing a strictly convex quadratic function of any number of variables. Based on the non-monotone line search techniques presented by Grippo et. al [11], Raydan [12] extended the BB method to solve large-scale unconstrained minimization problem. La Cruz and Raydan [13], presented a nonmonotone BB method to solve large-scale non-linear systems of equations. A global convergence result was obtained based on the variation of Grippo's non-monotone line search strategy which requires the computation of the first derivative. Motivated by the idea of La Cruz and Raydan, La Cruz et. al, [14], presented a globally convergent derivative-free BB algorithm using a combination of the Grippo's and Li-Fukushima's [15] line search strategies. Some modifications of BB method for solving unconstrained optimization can be found in [16, 17, 18], and reference therein. Nevertheless, because of its simplicity, efficiency and extremely low memory requirements, BB method interested many researchers. Applying it to solve non-linear systems of equations with a Jacobian that is not necessarily symmetric is a good research study.

Being a matrix-free approach, BB algorithm is capable of solving large scale unconstrained optimization problems. Motivated by the work of La Cruz et. al [14], in this paper we proposed a locally superlinearly BB-like method for solving large-scale non-linear systems of equations.

The remaining part of this paper is organized as follows. In section 2 we described the proposed method and its algorithm. The local superlinear convergence is established in section 3 and numerical results are reported in section 4. Throughout the paper $\|.\|$ stands for the Euclidean norm.

2. DESCRIPTION OF THE METHOD

We begin by describing the BB method for unconstrained minimization problem

$$\min_{x\in\mathbb{R}^n} f(x), \tag{6}$$

where $f:\mathbb{R}^n \to \mathbb{R}$. It is iteratively defined by

$$x_{k+1} = x_k - \sigma_k g(x_k), \qquad k = 0, 1, 2, ... \tag{7}$$

where $g(x_k) = \nabla f(x_k)$ and the scalar σ_k is given by

$$\text{either} \qquad \sigma_k = \frac{s_{k-1}^T s_{k-1}}{s_{k-1}^T y_{k-1}} \tag{8}$$

$$\text{or} \qquad \sigma_k = \frac{s_{k-1}^T y_{k-1}}{y_{k-1}^T y_{k-1}}, \tag{9}$$

where $s_{k-1} = x_k - x_{k-1}$, and $y_{k-1} = g(x_k) - g(x_{k-1}),\ k = 1, 2, 3, \ldots$.

If $F:\mathbb{R}^n \to \mathbb{R}^n$ is the gradient of some continuously differentiable function $f:\mathbb{R}^n \to \mathbb{R}$ then Eq. (1) is the first order necessary optimality condition of the unconstrained optimization problem (6).

Our aim is to solve problems of type Eq. (1), hence we choose to approximate the inverse Broyden matrix B_k^{-1} of Eq. (2) by $\sigma_k I$, where I is the identity matrix, then we obtain the new iterative process

$$x_{k+1} = x_k - \sigma_k F(x_k). \tag{10}$$

To obtain the parameter σ_k, let us consider a second order Taylor expansion of the residual $F(x_{k-1})$ about the iterate x_k:

$$F(x_{k-1}) \simeq F(x_k) - F'(x_k)s_{k-1}$$

which implies

$$F'(x_k)s_{k-1} \simeq F(x_k) - F(x_{k-1}).$$

Using B_k as the approximation of the Jacobian matrix $F'(x_k)$, we have the quasi-Newton condition

$$B_k s_{k-1} = \hat{y}_{k-1}, \tag{11}$$

where $\hat{y}_{k-1} = F(x_k) - F(x_{k-1})$.

Multiplying s_{k-1}^T to both sides of (11) and imposing a scalar approximation on B_k, say $B_k \simeq \alpha_k I$, where I is an $n \times n$ identity matrix, we have

$$\alpha_k = \frac{s_{k-1}^T \hat{y}_{k-1}}{s_{k-1}^T s_{k-1}}$$

.

Thus, if $s_{k-1}^T \hat{y}_{k-1} \neq 0$, then $\sigma_k = \alpha_k^{-1}$ is well-defined so that it can be used in (10) to solve (1).

We now present the BB-like algorithm.

Algorithm (BB-like Algorithm)

Step 0 Given $x_0 \in D \subset \mathbb{R}^n$, stopping tolerance $\epsilon > 0$, $\sigma_{min}, \sigma_{max} > 0$. Set $\sigma_0 = 1$.
For $k = 0$,
Step 1 Compute $\|F(x_0)\|$, if $\|F(x_0)\| \leq \epsilon$, stop.
Step 2 Compute $x_1 = x_0 - \sigma_0 F(x_0)$

For $k \geq 1$
Step 3 Compute

$$\sigma_k = \min\left\{\max\left\{\frac{s_{k-1}^T s_{k-1}}{s_{k-1}^T \hat{y}_{k-1}}, \sigma_{min}\right\}, \sigma_{max}\right\} \tag{12}$$

where $s_{k-1} = x_k - x_{k-1}$ and $\hat{y}_{k-1} = F(x_k) - F(x_{k-1})$
Step 4 Compute $x_{k+1} = x_k - \sigma_k F(x_k)$

Step 5 Compute $\|F(x_k)\|$, if $\|F(x_k)\| \leq \epsilon$, stop.
Step 6 Set $k = k + 1$ and goto Step 3.

Remark

1. The reasons behind computing σ_k using (12) in Step 3 are
   i. to avoid very small or large value of σ_k, which may cause unwanted uphill or downhill step respectively;
   ii. to ensure that the sequence $\{\sigma_k\}$ is uniformly bounded for each k. It is clear that $\sigma_k \leq \sigma_{max}$ $\forall k$.

3. CONVERGENCE ANALYSIS

In this section we present the local superlinear convergence of our proposed Barzilai-Borwein-like method for large-scale non-linear systems of equations. The following results will be useful in proving the main theorems of this section.

We consider an iteration of the form

$$x_{k+1} = G(x_k), \tag{13}$$

where G is a *fixed-point* map.

Definition 1:[19] Let $\Lambda \subset \mathbb{R}^n$ and $H : \Lambda \to \mathbb{R}^m$ be a map. Then H is Lipschitz continuous on Λ with Lipschitz constant γ if

$$\|H(x) - H(y)\| \leq \gamma\|x - y\|, \forall x, y \in \Lambda. \tag{14}$$

Definition 2:[19] Let $\Lambda \subset \mathbb{R}^n$. $G : \Lambda \to \mathbb{R}^n$ is a contraction mapping on Λ if G is Lipschitz continuous on Λ with Lipschitz constant $\gamma < 1$.

Definition 3:[**q-superlinearly** Convergence][19] 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 1:[19] Let $G : \Lambda \to \mathbb{R}^n$ be a contraction mapping on open and convex set Λ such that $G(x) \in \Lambda, \forall x \in \Lambda$. Then G has a unique fixed point $x^* \in \Lambda$ and the iteration defined by (13) converges to x^* for all $x_0 \in \Lambda$.

Lemma 2:[1] Let $F : \mathbb{R}^n \to \mathbb{R}^n$ be continuously differentiable in the open convex set $D \subset \mathbb{R}^n, x \in D$, and let F' be Lipschitz continuous at x in the neighborhood D with Lipschitz constant γ. Then for any $x + p \in D$,

$$\|F(x + p) - F(x) - F'(x)p\| \leq \frac{\gamma}{2}\|p\|^2.$$

Given the Euclidean norm $\|.\|$, let $B_\delta(x_0) = \{x \in \mathbb{R}^n : \|x - x_0\| < \delta\}$ be the open neighborhood of radius δ around x_0. In Theorem 1 and 2 below, we prove the convergence and the rate of convergence of our scheme respectively.

Theorem 1: Let $F : D \subset \mathbb{R}^n \to \mathbb{R}^n$ be continuously differentiable in an open convex set $D \subset \mathbb{R}^n$. Suppose that $\exists x^* \in \mathbb{R}^n$ and $\delta > 0$, such that $B_\delta(x^*) \subset D,\ F(x^*) = 0$. Then for all $x_0 \in B_\delta(x^*)$, the sequence $\{x_k\}$ generated by (10) is well-define and converges to x^*.

Proof: Let $x_0 \in B_\delta(x^*)$. Define a contraction map $G : D \subset \mathbb{R}^n \to \mathbb{R}^n$ by

$$G(x_k) = x_{k+1}.$$

Then x^* is a fixed point of G.

For $k = 0, 1, 2, ...,$ let $x_k \in D$.

$$\begin{aligned}
\|G(x_k) - x^*\| &= \|G(x_k) - G(x^*)\| \\
&\leq \gamma \|x_k - x^*\| \\
&= \gamma \|G(x_{k-1}) - G(x^*)\| \\
&\leq \gamma^2 \|x_{k-1} - x^*\| \\
&= \dots \\
&\leq \dots \\
&= \dots \\
&\leq \gamma^{k+1} \|x_0 - x^*\|.
\end{aligned}$$

So,

$$\begin{aligned}
\|G(x_k) - x^*\| &\leq \gamma^{k+1} \delta \\
&\leq \delta, \qquad \text{since } \gamma < 1.
\end{aligned}$$

Thus, $G(x_k) \in B_\delta(x^*) \subset D$. Hence, for any point $x_k \in D, x_{k+1} = G(x_k) \in D$ meaning that (10) is well-define.

All the hypothesis of Lemma 1 are now satisfied, so the sequence $\{x_k\}$ defined by (10) converges to a unique fixed point x^* of G for any $x_0 \in B_\delta(x^*)$.

Theorem 2: Let $F : D \subset \mathbb{R}^n \to \mathbb{R}^n$ be continuously differentiable in an open convex set $D \subset \mathbb{R}^n$. Suppose that $\exists x^* \in \mathbb{R}^n$ and $\delta > 0$, such that $B_\delta(x^*) \subset D,\ F(x^*) = 0$. Let F' be a contraction mapping on D. If $\|I - \sigma_k F'(x^*)\| < \gamma^k$ and $\exists \beta > 0$ such that $|\sigma_k| < \beta$, for each $k = 0, 1, 2, ...,$ then the sequence $\{x_k\}$ generated by (10) converges superlinearly to x^*.

Proof: Let $\beta > 0$. From (10) we have,

$$\begin{aligned} x_{k+1} - x^* &= x_k - x^* - \sigma_k(F(x_k) - F(x^*)) \\ &= x_k - x^* - \sigma_k(F'(x^*)(x_k - x^*) - F'(x^*)(x_k - x^*) \\ &\quad + F(x_k) - F(x^*)) \\ &= (I - \sigma_k F'(x^*))(x_k - x^*) - \sigma_k(-F'(x^*)(x_k - x^*) \\ &\quad + F(x_k) - F(x^*)). \end{aligned}$$

Then, using the theorem hypothesis and Lemma 2, we obtain

$$\begin{aligned} \|x_{k+1} - x^*\| &\leq \|I - \sigma_k F'(x^*)\| \|(x_k - x^*)\| \\ &\quad + \|\sigma_k(-F'(x^*)(x_k - x^*) + F(x_k) - F(x^*))\| \\ &\leq \gamma^k \|x_k - x*\| + \frac{\beta\gamma}{2}\|x_k - x^*\|^2. \end{aligned}$$

Dividing through by $\|x_k - x^*\|$, and taking limit as k tends to infinity yields,

$$\lim_{k\to\infty} \frac{\|x_{k+1} - x^*\|}{\|x_k - x^*\|} \leq \lim_{k\to\infty} \gamma^k + \frac{\beta\gamma}{2} \lim_{k\to\infty} \|x_k - x^*\|.$$

Since by Theorem 1 $\{x_k\}$ converges to x^*, $\lim_{k\to\infty} \|x_k - x^*\| = 0$, this proves that

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

4. NUMERICAL RESULTS

In this section we compare the performance of the proposed BB-like method (BBLM) with that of a matrix-free method named Diagonal Broyden-like method (DBLM) [8]. We used ten test functions of five instances of dimension $n = 100, 1000, 10000, 100000, 1000000$. [14]. This makes a total of fifty problems. In Table 1 we present results on the following information: the number of iterations (Iter) needed to converge to an approximate solution, the CPU time (in seconds) and the norm of the function at the approximate solution $\|F(x)\|$. Efficiency comparisons were made using the performance profile introduced by Dolan and More [20]. 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 second) reaches 1000, but no x_k satisfying $\|F(\mathrm{x}_k)\| \leq 10^{-8}$ is obtained. We implemented the two methods (DBLM and BBLM) using MATLAB R2010a and *tic- toc* command is used for reporting the CPU time. All computations were carried out on a PC with

Intel COREi3 processor with 4GB of RAM and CPU 2.30GHz.

The test functions $F(x) = (f_1(x), f_2(x), ..., f_n(x))^T$ are listed as follows:

Problem 1 Exponential function 1 [14]

$$f_1(x) = e^{x_1-1} - 1,$$
$$f_i(x) = i(e^{x_i-1} - x_i), \; i = 2, 3, ..., n.$$

Initial guess $x_0 = (0.5, 0.5, ..., 0.5)^T$

Problem 2 Logarithmic function [14]

$$f_i(x) = ln(x_i + 1) - \frac{x_i}{n}, \qquad i = 1, 2, 3, ..., n.$$

Initial guess $x_0 = (1, 1, ..., 1)^T$

Problem 3 Linear function-full rank [14]

$$f_i(x) = x_i - \frac{2}{n}\sum_{j=1}^{n} x_j + 1, \; i = 1, 2, ..., n.$$

Initial guess $x_0 = (100, 100, ..., 100)^T$

Problem 4 Tridiagonal exponential problem [21]

$$f_1(x) = x_1 - e^{\cos(h(x_1+x_2))},$$
$$f_i(x) = x_i - e^{\cos(h(x_{i-1}+x_i+x_{i+1}))}, \; i = 2, 3, ..., n-1.$$
$$f_n(x) = x_n - e^{\cos(h(x_{n-1}+x_n))},$$
$$h = 1/(n+1).$$

Initial guess $x_0 = (1.5, 1.5, ..., 1.5)^T$

Problem 5 Tridiagonal system [22]

$$f_1(x) = 4(x_1 - x_2^2),$$
$$f_i(x) = 8x_i(x_i^2 - x_{i-1}) - 2(1 - x_i) + 4(x_i - x_{i+1}^2), \; i = 2, 3, ..., n-1.$$
$$f_n(x) = 8x_n(x_n^2 - x_{n-1}) - 2(1 - x_n).$$

Initial guess $x_0 = (12, 12, ..., 12)^T$

Problem 6 Broyden Tridiagonal system [23]

$$f_1(x) = (3 - 0.5x_1)x_1 - 2x_2 + 1,$$
$$f_i(x) = (3 - 0.5x_i)x_i - x_{i-1} - 2x_{i+1} + 1, \; i = 2, 3, ..., n-1.$$
$$f_n(x) = (3 - 0.5x_n)x_n - x_{n-1} + 1.$$

Initial guess $x_0 = (-1.25, -1.25, ..., -1.25)^T$

Problem 7 Trigonometric system [24]

$$f_i(x) = 5 - (l+1)(1 - \cos x_i) - \sin x_i - \sum_{j=5l+1}^{5l+5} \cos x_j,\ \ i = 1, 2, ..., n.$$

$$l = \text{div}(i-1, 5).$$

Initial guess $x_0 = (1/n, 1/n, ..., 1/n)^T$

Problem 8 Trigonometric function [14]

$$f_i(x) = 2\left(n + i(1 - \cos x_i) - \sin x_i - \sum_{j=1}^{n} \cos x_j\right)(2 \sin x_i - \cos x_i),$$

$$i = 1, 2, ..., n.$$

Initial guess $x_0 = \left(\frac{101}{100n}, \frac{101}{100n}, ..., \frac{101}{100n}\right)^T$

Problem 9 [8]

$$f_i(x) = ln(x_i) \cos(1 - (1 + x^T x)^2)^{-1}) e^{1-(1+x^T x)^2)^{-1}},\ \ i = 1, 2, ..., n.$$

Initial guess $x_0 = (2.5, 2.5, ..., 2.5)^T$

Problem 10 [7]

$$f_i(x) = (\cos(x_i) - 1)^2 - 1,\ \ i = 1, 2, 3, ..., n.$$

Initial guess $x_0 = (1, 1, ..., 1)^T$

The numerical results of the performance of DBLM and BBLM relative to number of iterations (Iter) and CPU time (Time) in Table 1, are interpreted in Figure 1 and Figure 2 respectively. In Figure 1 BBLM solve more than 70% while DBLM solves around 20% of the problems with a lower number of iterations. From Figure 2 it is easy to see that BBLM is more competitive than DBLM since it solve about 80% of the problems within shorter time. In short, BBLM solves and wins 58% and DBLM solves and wins 28% of the total tested problems.

4. CONCLUSIONS

We have proposed a derivative-free approach for solving non-linear systems of equations in which the inverse Jacobian matrix is approximated using a scalar multiple of identity σ_k which is obtained from the quasi-Newton equation. This approach is based on the well-known Barzilai and Borwein method for unconstrained optimization problems. Due to the simplicity of this approach, our

Table 1. Numerical comparison for problems 1 to 10

	DBLM			BBLM		
P(n)	**Iter**	**Time**	$\|\mathbf{F(x)}\|$	**Iter**	**Time**	$\|\mathbf{F(x)}\|$
1(100)	-	-	-	50	0.007461	6.6775×10^{-9}
1(1000)	-	-	-	90	0.028288	6.4682×10^{-9}
1(10000)	-	-	-	56	0.132716	2.8983×10^{-10}
1(100000)	-	-	-	56	0.883515	1.3339×10^{-10}
1(1000000)	-	-	-	44	9.447604	1.4989×10^{-10}
2(100)	6	0.004256	3.5178×10^{-11}	7	0.001343	9.2766×10^{-13}
2(1000)	5	0.004338	7.8063×10^{-11}	7	0.005542	1.6344×10^{-12}
2(10000)	5	0.027397	5.4546×10^{-10}	7	0.029785	4.8845×10^{-12}
2(100000)	5	0.197403	1.7603×10^{-10}	7	0.157910	1.5307×10^{-11}
2(1000000)	5	1.945043	9.1038×10^{-11}	7	1.595762	4.8406×10^{-11}
3(100)	169	0.040537	8.315×10^{-9}	2	0.001112	0
3(1000)	-	-	-	2	0.002127	0
3(10000)	-	-	-	2	0.007579	0
3(100000)	-	-	-	2	0.041075	1.7133×10^{-11}
3(1000000)	-	-	-	2	0.503012	0
4(100)	5	0.001270	9.5912×10^{-11}	4	0.001516	1.6132×10^{-10}
4(1000)	3	0.003427	8.6934×10^{-12}	3	0.002656	3.0497×10^{-14}
4(10000)	2	0.016084	4.1784×10^{-11}	2	0.02767	9.3684×10^{-12}
4(100000)	2	0.116230	0	2	0.065398	0
4(1000000)	2	1.412305	0	2	0.825733	0
5(100)	-	-	-	-	-	-
5(1000)	-	-	-	-	-	-
5(10000)	-	-	-	-	-	-
5(100000)	-	-	-	-	-	-
5(1000000)	-	-	-	-	-	-
6(100)	-	-	-	34	0.011190	6.1063×10^{-9}
6(1000)	-	-	-	35	0.038376	6.9540×10^{-9}
6(10000)	-	-	-	36	0.111198	7.0494×10^{-9}
6(100000)	-	-	-	47	0.994749	2.2734×10^{-9}
6(1000000)	-	-	-	60	18.361573	8.4602×10^{-9}
7(100)	260	0.127542	7.3902×10^{-9}	9	0.003651	3.0099×10^{-10}
7(1000)	-	-	-	9	0.017774	1.2031×10^{-9}
7(10000)	-	-	-	9	0.050137	3.9746×10^{-10}
7(100000)	-	-	-	9	0.235802	1.9211×10^{-10}
7(1000000)	-	-	-	9	2.791206	0
8(100)	-	-	-	16	0.016368	1.3745×10^{-10}
8(1000)	-	-	-	20	0.019402	3.1637×10^{-10}
8(10000)	-	-	-	24	0.12777	0
8(100000)	-	-	-	-	-	-
8(1000000)	-	-	-	-	-	-
9(100)	7	0.010642	1.3974×10^{-9}	8	0.004539	1.1323×10^{-13}
9(1000)	6	0.007679	8.7338×10^{-10}	8	0.007523	3.6513×10^{-13}
9(10000)	6	0.074735	4.3898×10^{-11}	8	0.064629	1.1546×10^{-12}
9(100000)	6	0.327127	2.1065×10^{-13}	8	0.242533	3.6513×10^{-12}
9(1000000)	6	3.521195	4.4409×10^{-13}	8	2.779271	1.1546×10^{-11}
10(100)	64	0.026960	6.3805×10^{-9}	6	0.002842	4.3451×10^{-9}
10(1000)	203	0.110147	8.9837×10^{-9}	7	0.005214	1.4043×10^{-14}
10(10000)	649	1.985304	9.6267×10^{-9}	7	0.055379	4.4409×10^{-14}
10(100000)	-	-	-	7	0.113214	1.4043×10^{-13}
10(1000000)	-	-	-	7	1.120738	4.4409×10^{-13}

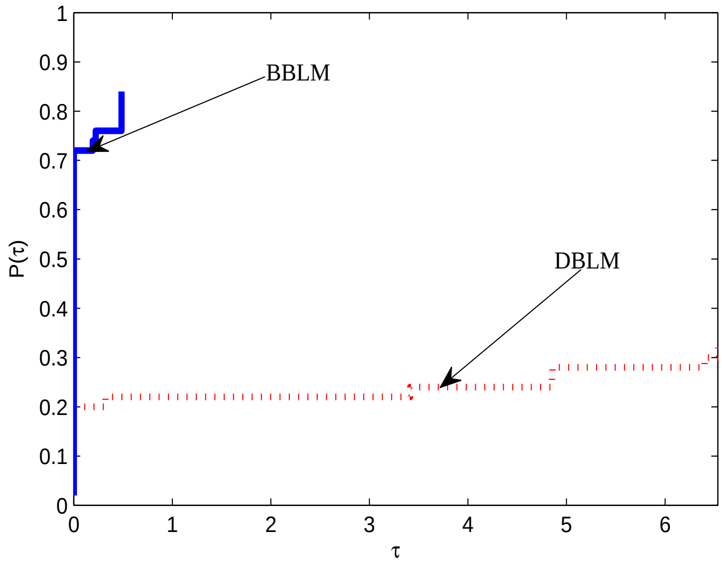

FIGURE 1. Performance profile of DBLM and BBLM methods with respect to number of iterations for problem 1-10

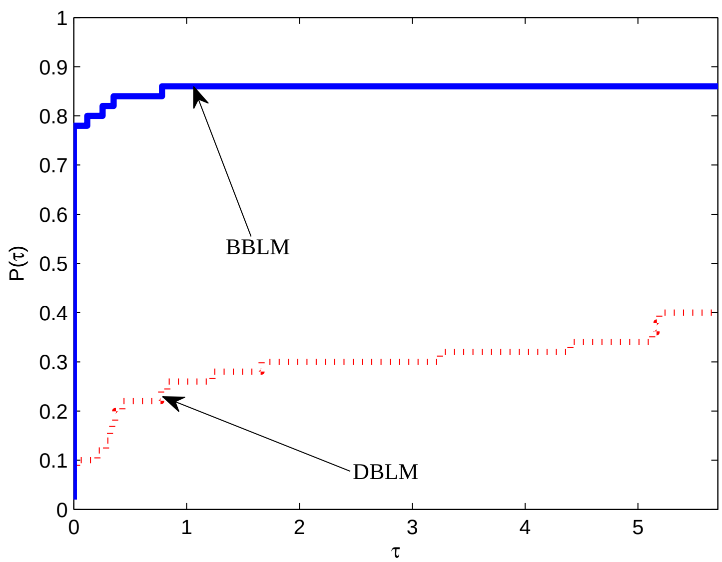

FIGURE 2. Performance profile of DBLM and BBLM methods with respect to CPU time for problem 1-10

proposed method can be used for solving large-scale non-linear systems.

Under some assumptions, the convergence rate of the new BB-like method was shown to be superlinear. Numerical results show that the proposed method is competitive to similar method for large-scale problems. It is worth mentioning that the convergence results of the BB-like algorithm presented in this paper is different from the one given by La Cruz et. al. [14].

ACKNOWLEDGEMENTS

The author would like to thank Professor Mehiddin Al-Baali of Sultan Qaboos University, Oman, whose comments improved the original version of this manuscript.

References

[1] J. E. Dennis, R. B. Schnabel, Junior, Numerical Methods for Unconstrained Optimization and Nonlinear Equations, Prentice-Hall, 1983.
[2] C. G. Broyden, A class of methods for solving nonlinear simultaneous equations, Math Comput 19 (1965) 577–593.
[3] J. E. Dennis, J. J. More, A characterization of superlinear convergence and its application to quasi-newton methods, Math Comput 28 (1974) 549–560.
[4] M. Al-Baali, E. Spedicato, F. Maggioni, Broyden's quasi-newton methods for a nonlinear system of equations and unconstrained optimization: a review and open problems, Optimization Methods and Software 29 (5) (2014) 937–954.
[5] J. M. Martinéz, Practical quasi-newton methods for solving nonlinear systems, J Comput App Math 124 (2000) 97–121.
[6] A. Dhamacharoen, An efficient hybrid method for solving systems of nonlinear equations, J. Comput. Appl. Math. 263 (2014) 59–68.
[7] M. Hassan, M. Y. Waziri, On broyden-like update via some quadratures for solving nonlinear systems of equations, Turkish Journal of Mathematics 39 (3) (2015) 335–345.
[8] M. Waziri, W. Leong, M. Hassan, Diagonal broyden-like method for large-scale systems of nonlinear equations, Malaysian Journal of Mathematical Sciences 6 (1) (2012) 59–73.
[9] B. J. M. Barzilai, Jonathan, Two-point step size gradient methods, IMA Journal of Numerical Analysis 8 (1) (1988) 141–148.
[10] M. Raydan, On the barzilai and borwein choice of steplength for the gradient method, IMA Journal of Numerical Analysis 13 (3) (1993) 321–326.
[11] L. F. L. S. Grippo, Luigi, A nonmonotone line search technique for newton's method, SIAM Journal on Numerical Analysis 23 (4) (1986) 707–716.
[12] M. Raydan, The barzilai and borwein gradient method for the large scale unconstrained minimization problem, SIAM Journal on Optimization 7 (1) (1997) 26–33.

[13] R. M. Cruz, William La, Nonmonotone spectral methods for large-scale nonlinear systems, Optimization Methods and Software 18 (5) (2003) 583–599.

[14] M. J. M. R. M. La Cruz, William, Spectral residual method without gradient information for solving large-scale nonlinear systems: theory and experiments.

[15] F. M. Li, Dong-Hui, A derivative-free line search and global convergence of broyden-like method for nonlinear equations, Optimization Methods and Software 13 (3) (2000) 181–201.

[16] Y.-H. Dai, A new analysis on the barzilai-borwein gradient method, Journal of the Operations Research Society of China 1 (2) (2013) 187–198.

[17] S. Setzer, G. Steidl, J. Morgenthaler, A cyclic projected gradient method, Computational Optimization and Applications 54 (2) (2013) 417–440.

[18] Y. Xiao, Q. Wang, D. Wang, Notes on the dai–yuan–yuan modified spectral gradient method, Journal of computational and applied mathematics 234 (10) (2010) 2986–2992.

[19] C. T. Kelly, Iterative Methods for Linear and Nonlinear Equations, SIAM, 1995.

[20] M. J. J. Dolan, E. D., Benchmarking optimization software with performance profiles, Math. Program., Ser 91 (2002) 201–213.

[21] Y. Bing, G. Lin, An efficient implementation of merrill's method for sparse or partially separable systems of nonlinear equations, SIAM Journal on Optimization 1 (2) (1991) 206–221.

[22] G. Li, Successive column correction algorithms for solving sparse nonlinear systems of equations, Mathematical Programming 43 (1-3) (1989) 187–207.

[23] M. J. M. M. A. C. Gomes-Ruggiero, Márcia A, Comparing algorithms for solving sparse nonlinear systems of equations, SIAM Journal on Scientific and Statistical Computing 13 (2) (1992) 459–483.

[24] P. L. Toint, Some numerical results using a sparse matrix updating formula in unconstrained optimization, Mathematics of Computation 32 (143) (1978) 839–851.

DEPARTMENT OF MATHEMATICAL SCIENCES, BAYERO UNIVERSITY, KANO, NIGERIA

E-mail addresses: hmuhammad.mth@buk.edu.ng