Yokohama Publishers
ISSN 1349-8169 ONLINE JOURNAL

Pacific Journal of Optimization
Vol. 21, No. 1, pp. 89–106, 2025

A BFGS METHOD USING INEXACT GRADIENT FOR GENERAL NONLINEAR EQUATIONS

Weijun Zhou and Li Zhang

Abstract: A globally and superlinearly convergent BFGS method is introduced to solve general nonlinear equations without computing exact gradient. Compared with existing Gauss-Newton-based BFGS type methods, the proposed method does not require conditions such as symmetry on the underlying function. Moreover, it can be suitably adjusted to solve nonlinear least squares problems and still guarantee global convergence. Some numerical results are reported to show its efficiency.

Key words: *general nonlinear equations, BFGS, line search, global convergence*

Mathematics Subject Classification: *65K05, 90C30*

1 Introduction

Let $F : R^n \to R^n$ be a continuously differentiable mapping. We consider numerical methods for solving general nonlinear equations

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

When the Jacobian $J(x) = F'(x)$ can be used, Newton methods, Gauss-Newton methods and Levenberg-Marquardt methods are very attractive [11, 23]. When n is large and F has special structure, derivative-free methods and subspace methods are always adopted [8, 18, 19, 25].

In this paper we focus our attention on BFGS type quasi-Newton methods with line search for the general system which has no special structure. BFGS type methods are efficient for solving nonlinear equations and optimization problems since they possess locally superlinear convergence properties and need not compute the Jacobian or the Hessian [3, 9, 11, 14, 15, 18]. However, global convergence of BFGS type methods for nonlinear equations often require special structure such as symmetry or very strong assumption conditions [2, 5, 6, 12, 13, 16, 17, 21, 22, 24].

To our knowledge, the first global convergence result of BFGS type methods for nonlinear equations is due to Li and Fukushima [6], where they introduced a Gauss-Newton-based BFGS method (GN-BFGS), which has been extended to solve symmetric nonlinear least squares [21]. But both methods need the assumption that the system is symmetric (i.e., $J(x) = J(x)^T$), which restricts their applications. The aim of this paper is to generalize the GN-BFGS method such that it can be suitable to solve general nonlinear system without computing exact Jacobian of the system or exact gradient.

 DOI: https://doi.org/10.61208/pjo-2023-027

This paper is organized as follows. In the next section, we first present the method in detail. Then we prove its global convergence under some conditions. In Section 3, we show superlinear convergence of the proposed method. In Section 4, we report some numerical results. Throughout the paper, we denote $F_k = F(x_k), J_k = J(x_k), s_k = x_{k+1} - x_k = \alpha_k d_k$ and $\|\cdot\|$ is the 2-norm.

2 Algorithm and Global Convergence

In this section, we first illustrate our approach which is mainly based on the following consideration. In [6], the GN-BFGS method for symmetric nonlinear equations produces the search direction d_k by computing the linear equations

$$B_k d = -\frac{F(x_k + \lambda_{k-1}F_k) - F(x_k)}{\lambda_{k-1}},$$

where $\lambda_{k-1} > 0$ is a stepsize and the iterative matrix B_k is updated by the BFGS formula

$$B_{k+1} = B_k - \frac{B_k s_k s_k^T B_k}{s_k^T B_k s_k} + \frac{\gamma_k \gamma_k^T}{\gamma_k^T s_k}$$

with $\gamma_k = F(x_k + \delta_k) - F(x_k), \delta_k = F(x_{k+1}) - F(x_k)$. Let us define

$$f(x) \triangleq \frac{1}{2}\|F(x)\|^2. \tag{2.1}$$

When $\|F_k\|$ is small and the Jacobian is symmetric, $\frac{F(x_k+\lambda_{k-1}F_k)-F(x_k)}{\lambda_{k-1}} \approx \nabla f(x_k)$, which shows that this is a method based on inexact gradient. Moreover, if $J(x)$ satisfies Lipschitz condition with Lipschitz constant $\bar{L}$, then the symmetry of the Jacobian implies

$$\begin{aligned} \left\|\frac{F(x_k + \lambda_{k-1}F_k) - F(x_k)}{\lambda_{k-1}} - \nabla f(x_k)\right\| &= \left\|\int_0^1 \big(J(x_k + t\lambda_{k-1}F_k) - J(x_k)\big)F_k dt\right\| \\ &\le \bar{L}\lambda_{k-1}\|F_k\|^2, \end{aligned} \tag{2.2}$$

which gives an error estimation between $\frac{F(x_k+\lambda_{k-1}F_k)-F(x_k)}{\lambda_{k-1}}$ and $\nabla f(x_k)$.

Thus for general nonlinear equations, the key is to construct a suitable term which approximates $\nabla f(x_k)$. In this case, it is clear that the term $\frac{F(x_k+\lambda_{k-1}F_k)-F(x_k)}{\lambda_{k-1}}$ is no longer a suitable approximation of $\nabla f(x_k)$. But we note that when $\|F_k\|$ is small,

$$\frac{f(x_k + \alpha\|F_k\|^2 e_i) - f(x_k)}{\alpha\|F_k\|^2} \approx \frac{\partial f(x_k)}{\partial x_i},$$

where e_i is the i-th column of the identity matrix and $\alpha > 0$ is a parameter. Hence the term

$$g_k(\alpha) \triangleq \frac{1}{\|F_k\|^2} \begin{pmatrix} \frac{f(x_k+\alpha\|F_k\|^2 e_1)-f(x_k)}{\alpha} \\ \frac{f(x_k+\alpha\|F_k\|^2 e_2)-f(x_k)}{\alpha} \\ \vdots \\ \frac{f(x_k+\alpha\|F_k\|^2 e_n)-f(x_k)}{\alpha} \end{pmatrix} \tag{2.3}$$

is an approximation of $\nabla f(x_k)$, which also can be seen from the following equality

$$\lim_{\alpha \to 0^+} g_k(\alpha) = \nabla f(x_k). \tag{2.4}$$

Moreover, if $\nabla f(x)$ satisfies Lipschitz condition with Lipschitz constant $\bar{L}_1$, then

$$\begin{aligned} \|g_k(\alpha) - \nabla f(x_k)\| &= \left\| \begin{pmatrix} \int_0^1 \left(\nabla f(x_k + t\alpha\|F_k\|^2 e_1) - \nabla f(x_k)\right)^T e_1 dt \\ \vdots \\ \int_0^1 \left(\nabla f(x_k + t\alpha\|F_k\|^2 e_n) - \nabla f(x_k)\right)^T e_n dt \end{pmatrix} \right\| \\ &\le n\bar{L}_1 \alpha \|F_k\|^2. \end{aligned} \tag{2.5}$$

This is the main reason of choosing $g_k(\alpha)$ instead of choosing the term

$$\frac{1}{\|F_k\|} \begin{pmatrix} \frac{f(x_k+\alpha\|F_k\|e_1)-f(x_k)}{\alpha} \\ \frac{f(x_k+\alpha\|F_k\|e_2)-f(x_k)}{\alpha} \\ \vdots \\ \frac{f(x_k+\alpha\|F_k\|e_n)-f(x_k)}{\alpha} \end{pmatrix}$$

as an approximation of $\nabla f(x_k)$ since the approximate precision between $g_k(\alpha)$ and $\nabla f(x_k)$ given by (2.5) in the general nonlinear case is the same as that of (2.2) in the symmetric nonlinear case.

Assume that we have a parameter α_{k-1} at the moment, then we get the search direction d_k by letting it be the solution of the following linear equations:

$$B_k d = -g_k, \tag{2.6}$$

where

$$g_k \triangleq g_k(\alpha_{k-1}). \tag{2.7}$$

Moreover, we adopt the line search proposed by Li and Fukushima [6] to compute the next stepsize α_k. Let $\sigma_1 > 0$ and $\sigma_2 > 0$ be two given constants and $\{\eta_k\}$ be a positive sequence satisfying

$$\sum_{k=0}^{\infty} \eta_k \le \eta < \infty, \tag{2.8}$$

where η is a positive constant. We compute $\alpha_k = \max\{1, \rho, \rho^2, \cdots\}$ such that

$$f(x_k + \alpha d_k) \le f(x_k) - \sigma_1\|\alpha d_k\|^2 - \sigma_2\|\alpha F_k\|^2 + \eta_k f(x_k), \tag{2.9}$$

where $\rho \in (0,1)$ is a constant. It is clear that the line search (2.9) is well-defined.

Therefore, we are ready for presenting the following BFGS method for solving (1.1).

Algorithm 2.1

Step 0. Choose a starting point $x_0 \in R^n$, an initial symmetric positive definite matrix $B_0 \in R^{n\times n}$, a positive sequence $\{\eta_k\}$ satisfying (2.8), and several constants $\alpha_{-1} > 0$, $\sigma_1 > 0, \sigma_2 \in (0, \frac{1}{2})$, $\mu > 0$ and $\rho_0, \rho \in (0,1)$. Let $k := 0$.

Step 1. Compute d_k by (2.6).

Step 2. If

$$\|F(x_k + d_k)\| \le \rho_0\|F(x_k)\|, \tag{2.10}$$

then set $\alpha_k = 1$. Otherwise, compute α_k by the line search (2.9).

Step 3. Set $x_{k+1} = x_k + \alpha_k d_k$.

Step 4. Update B_k by the following rule,

$$B_{k+1} = \begin{cases} B_k - \frac{B_k s_k s_k^T B_k}{s_k^T B_k s_k} + \frac{y_k y_k^T}{y_k^T s_k}, & \text{if} \quad \frac{y_k^T s_k}{\|s_k\|^2} \geq \mu \|F_k\|, \\ B_k, & \text{otherwise}, \end{cases} \tag{2.11}$$

where

$$y_k \triangleq \bar{g}_{k+1} - g_k \tag{2.12}$$

and

$$\bar{g}_{k+1} \triangleq g_{k+1}(\alpha_{k-1}) = \frac{1}{\|F_{k+1}\|^2} \begin{pmatrix} \frac{f(x_{k+1}+\alpha_{k-1}\|F_{k+1}\|^2 e_1) - f(x_{k+1})}{\alpha_{k-1}} \\ \frac{f(x_{k+1}+\alpha_{k-1}\|F_{k+1}\|^2 e_2) - f(x_{k+1})}{\alpha_{k-1}} \\ \vdots \\ \frac{f(x_{k+1}+\alpha_{k-1}\|F_{k+1}\|^2 e_n) - f(x_{k+1})}{\alpha_{k-1}} \end{pmatrix}. \tag{2.13}$$

Step 5. Let $k := k+1$ and go to Step 1.

Remark 2.1

(i) In Step 4, we use the cautious BFGS formula proposed by Li and Fukushima [7] where it was used to solve nonconvex unconstrained minimization problems. The update rule (2.11) ensures that the iterative matrix sequence $\{B_k\}$ is symmetric and positive definite.

(ii) In the global convergence for some methods such as the BFGS methods in [7, 20], it need the Lipschitz assumption on the gradient, which implies $\|\nabla f(x_{k+1}) - \nabla f(x_k)\| \leq \bar{L}_2 \|s_k\|$ for some constant $\bar{L}_2$. From (2.20), we can see later that $\|\bar{g}_{k+1} - g_k\|$ satisfies the similar condition. This is the reason of using $\bar{g}_{k+1}$ instead of g_{k+1} in (2.12).

(iii) From (2.9) and (2.10), if $\alpha_k \neq 1$, then $\alpha'_k = \frac{\alpha_k}{\rho}$ does not satisfy (2.9), that is,

$$f(x_k + \alpha'_k d_k) > f(x_k) - \sigma_1 \|\alpha'_k d_k\|^2 - \sigma_2 \|\alpha'_k F_k\|^2. \tag{2.14}$$

(iv) By (2.8) and (2.9), it is easy to get

$$\prod_{i=0}^{\infty} (1 + \eta_i) \leq e^{\eta}, \tag{2.15}$$

and $f(x_{k+1}) \leq (1 + \eta_k) f(x_k)$. Then by Lemma 3.3 in [3] that $\{f(x_k)\}$ converges.

From now on, we begin to investigate global convergence property of Algorithm 2.1. To this end, we make some assumptions as follows.

Assumption 2.1

(i) The level set $\Omega = \{x | \ f(x) \leq e^{\eta} f(x_0)\}$ is bounded.

(ii) In some neighbourhood Ω_1 of Ω, $J(x)$ is Lipschitz continuous, that is, there exists a positive constant L such that

$$\|J(x) - J(y)\| \leq L \|x - y\|, \quad \forall x, y \in \Omega_1. \tag{2.16}$$

It is clear that the sequence $\{x_k\} \subset \Omega$. Moreover, Assumption 2.1 implies that there exist positive constants M_1, M_2, L_1 and L_2 such that

$$\|J(x)\| \leq M_1, \quad \|F(x)\| \leq M_2, \quad \forall x \in \Omega_1, \tag{2.17}$$

$$\|F(x) - F(y)\| \leq L_1 \|x - y\|, \quad \|\nabla f(x) - \nabla f(y)\| \leq L_2 \|x - y\|, \quad \forall x, y \in \Omega_1. \tag{2.18}$$

Lemma 2.1. *Let Assumption 2.1 hold and the sequence $\{x_k\}$ be generated by Algorithm 2.1. If (2.10) holds for infinite k, then $\lim_{k\to\infty}\|F_k\|=0$. Otherwise, we have*

$$\sum_{k=0}^{\infty}\|s_k\|^2<\infty,\quad \sum_{k=0}^{\infty}\|\alpha_k F_k\|^2<\infty.$$

Proof. If (2.10) holds for infinite k, then it is obvious that $\lim_{k\to\infty}\|F_k\|=0$. If (2.10) holds only for finite k, then by (2.9), we have

$$\sum_{i=0}^{k}\left(\sigma_1\|s_i\|^2+\sigma_2\|\alpha_i F_i\|^2\right)\le f(x_0)-f(x_{k+1})+\sum_{i=0}^{k}\eta_i f(x_i).$$

Since $f(x_{k+1})\ge 0$ and $\{f(x_k)\}$ convergence, there exists a constant $\bar{M}>0$ such that

$$\sum_{i=0}^{k}\left(\sigma_1\|s_i\|^2+\sigma_2\|\alpha_i F_i\|^2\right)\le f(x_0)+\sum_{i=0}^{k}\bar{M}\eta_i.$$

This and (2.8) yield the conclusion. □

Now we assume (2.10) holds for finite k. Then Lemma 2.1 shows

$$\lim_{k\to\infty}\|s_k\|=0,\quad \lim_{k\to\infty}\alpha_k\|F_k\|=0. \tag{2.19}$$

Moreover, (2.13) and (2.18) yield

$$\begin{aligned}
\|y_k\| &= \|\bar{g}_{k+1}-g_k\|\\
&= \left\|\begin{pmatrix}\int_0^1\left(\nabla f(x_{k+1}+t\alpha_{k-1}\|F_{k+1}\|^2e_1)-\nabla f(x_k+t\alpha_{k-1}\|F_k\|^2e_1)\right)^T e_1dt\\ \vdots\\ \int_0^1\left(\nabla f(x_{k+1}+t\alpha_{k-1}\|F_{k+1}\|^2e_n)-\nabla f(x_k+t\alpha_{k-1}\|F_k\|^2e_n)\right)^T e_ndt\end{pmatrix}\right\|\\
&\le nL_2(\|x_{k+1}-x_k\|+\alpha_{k-1}|\|F_{k+1}\|^2-\|F_k\|^2|)\\
&\le nL_2\big(\|x_{k+1}-x_k\|+\alpha_{k-1}(\|F_{k+1}\|+\|F_k\|)(|\|F_{k+1}\|-\|F_k\||)\big)\\
&\le nL_2(1+2L_1M_2)\|s_k\|,
\end{aligned} \tag{2.20}$$

where the last inequality follows from (2.17), (2.18) and the fact $\alpha_{k-1}\le 1$. Moreover, if there exists a positive constant τ_1 such that

$$\|\nabla f(x_k)\|\ge\tau_1 \tag{2.21}$$

holds for all k. Then there is a positive constant τ_2 satisfying

$$\|F_k\|\ge\tau_2,\quad \forall k\ge 0. \tag{2.22}$$

From the update rule (2.11), if B_k is updated by the BFGS formula, then

$$\frac{y_k^Ts_k}{\|s_k\|^2}\ge\mu\tau_2,\quad \frac{y_k^Ty_k}{y_k^Ts_k}\le\frac{(nL_2(1+2L_1M_2))^2\|s_k\|^2}{\mu\tau_2\|s_k\|^2}=\frac{(nL_2(1+2L_1M_2))^2}{\mu\tau_2}\triangleq C_1$$

hold for some positive constant C_1.

Lemma 2.2. *Let Assumption 2.1 hold. If (2.21) holds, then there are positive constants* $\beta_i, i = 1, 2, 3, 4$ *such that*

$$\beta_1\|s_i\| \le \|B_i s_i\| \le \beta_2\|s_i\|, \quad \beta_3\|s_i\|^2 \le s_i^T B_i s_i \le \beta_4\|s_i\|^2$$

hold for at least $\lceil \frac{|A_k|}{2} \rceil$ *many* $i \in A_k$*, where* $|A_k|$ *is the cardinality of the set* A_k *and*

$$A_k = \Big\{ i | \; i \le k, B_{i+1} = B_i - \frac{B_i s_i s_i^T B_i}{s_i^T B_i s_i} + \frac{y_i y_i^T}{y_i^T s_i} \Big\}.$$

Proof. By (2.11), we know that B_{k+1} is updated by the BFGS formula or $B_{k+1} = B_k$. Then we have

$$\begin{aligned} \psi(B_{k+1}) \quad &\le \quad \psi(B_0) + \big(C_1 - 1 - \ln(\mu\tau_2)\big)|A_k| \\ &\quad + \sum_{j \in A_k} \Big(\ln\cos^2\theta_j + 1 - \frac{q_j}{\cos^2\theta_j} + \ln\frac{q_j}{\cos^2\theta_j} \Big), \end{aligned}$$

where $\psi(B) = \mathrm{tr}(B) - \ln(\det(B)), q_j = \frac{s_j^T B_j s_j}{s_j^T s_j}, \cos\theta_j = \frac{s_j^T B_j s_j}{\|s_j\|\|B_j s_j\|}$. Thus using the same argument as that of Theorem 2.1 in [1], we obtain the conclusion. □

Since $s_k = \alpha_k d_k$, Lemma 2.2 and (2.6) imply that

$$\beta_1\|d_i\| \le \|B_i d_i\| = \|g_i\| \le \beta_2\|d_i\|, \quad \beta_3\|d_i\|^2 \le d_i^T B_i d_i = -d_i^T g_i \le \beta_4\|d_i\|^2 \tag{2.23}$$

hold for at least $\lceil \frac{|A_k|}{2} \rceil$ many $i \in A_k$.

The following result shows that Algorithm 2.1 converges globally.

Theorem 2.3. *Let Assumption 2.1 hold and the sequence* $\{x_k\}$ *be generated by Algorithm 2.1. Then we have*

$$\liminf_{k\to\infty} \|\nabla f(x_k)\| = 0. \tag{2.24}$$

Proof. We prove the theorem by contradiction. If (2.24) is not true, then (2.21)-(2.22) hold and (2.10) only holds for finite k.

(i) If $\limsup_{k\to\infty} \alpha_k > 0$, then we deduce from (2.19) that

$$\liminf_{k\to\infty} \|F_k\| = 0,$$

which contradicts (2.22).

(ii) If $\lim_{k\to\infty} \alpha_k = 0$, then (2.14) holds for $\alpha'_k = \frac{\alpha_k}{\rho}$. Moreover, from (2.5) and (2.21), there exists a positive constant τ_3 such that

$$\|g_k\| \ge \tau_3, \quad \forall k \ge 0. \tag{2.25}$$

Denote

$$\hat{K} \triangleq \bigcup_{k>0} A_k, \quad K \triangleq \{i|(2.23) \text{ holds}\}. \tag{2.26}$$

If $\hat{K}$ is finite, then the conclusion is clear. If $\hat{K}$ is infinite, then K is infinite. Since $\{x_k\}$ is bounded, then the sequence $\{g_k\}_{k\in K}$ and $\{d_k\}_{k\in K}$ are bounded. Without loss of generality, let the sequences $\{d_k\}_{k\in K}$ and $\{x_k\}_{k\in K}$ converge to d^* and x^*, respectively. Hence $\lim_{k\in K, k\to\infty} g_k = \nabla f(x^*)$. Let $k \to \infty$ with $k \in K$ in (2.14), then

$$\nabla f(x^*)^T d^* \ge 0. \tag{2.27}$$

By (2.6), we get $0 = d_k^T B_k d_k + g_k^T d_k$, which together with (2.23) implies $0 \geq \beta_3 \|d_k\|^2 + g_k^T d_k, \forall k \in K$. Let $k \to \infty$ with $k \in K$ in this inequality, we have

$$\nabla f(x^*)^T d^* \leq -\beta_3 \|d^*\|^2.$$

This and (2.27) show that $d^* = 0$. Since $g_k = -B_k d_k$, then from (2.23), we know

$$\lim_{k\in K, k\to\infty} \|g_k\| = \lim_{k\in K, k\to\infty} \|d_k\| = \|d^*\| = 0,$$

which contradicts with (2.25). This completes the proof. □

Remark 2.2 It is clear that Algorithm 2.1 can be applied to solving the following nonlinear least squares problem

$$\min f(x) \triangleq \frac{1}{2}\|F(x)\|^2, \tag{2.28}$$

where $F : R^n \to R^m$ is a general vector value mapping and n may be not equal to m. And from the previous analysis, it is easy to show that Algorithm 2.1 still converges globally for (2.28), which is also supported by the numerical results later.

3 Superlinear Convergence

In this section, we turn to discussing the superlinear convergence of Algorithm 2.1. To do this, we need the following assumptions.

Assumption 3.1

(I) The sequence $\{x_k\}$ converges to x^*, where $F(x^*) = 0$ and $J(x^*)$ is nonsingular.

(II) In some neighourhood Ω_2 of x^*, $\nabla^2 f$ is Lipschitz continuous, i.e., there exists a positive constant L_3 satisfying

$$\|\nabla^2 f(x) - \nabla^2 f(y)\| \leq L_3 \|x - y\|, \forall x, y \in \Omega_2. \tag{3.1}$$

Without loss of generality, we assume $\{x_k\} \subseteq \Omega_2$. Assumption (I) implies that $J(x)$ is uniformly nonsingular in Ω_2, that is, there is a constant $m_1 > 0$ such that

$$m_1\|d\| \leq \|J(x)d\|, \quad m_1\|d\| \leq \|J(x)^{-1}d\|, \quad \forall x \in \Omega_2, d \in R^n. \tag{3.2}$$

This shows

$$\|F_k\| \leq \frac{1}{m_1}\|\nabla f(x_k)\|. \tag{3.3}$$

Moreover, by (2.5), we know

$$\|\nabla f(x_k)\| \leq \|g_k\| + nL_2\alpha_{k-1}\|F_k\|^2. \tag{3.4}$$

Since $\lim_{k\to\infty} \|F_k\| = 0$ and $\alpha_{k-1} \leq 1$, from (3.3) and (3.4), there exists a constant $m_2 > 0$ such that for sufficiently large k,

$$\|F_k\| \leq m_2\|g_k\|. \tag{3.5}$$

This together with (2.23) implies that for any $k \in K$,

$$\|F_k\| \leq m_2\beta_2\|d_k\|. \tag{3.6}$$

Since $F_k - F(x^*) = F_k = \int_0^1 J(x^* + t(x_k - x^*))dt(x_k - x^*)$, hence from (3.2), there exist two constants $m_3 > 0$ and $m_4 > 0$ such that

$$m_3\|x_k - x^*\| \leq \|F_k\| \leq m_4\|x_k - x^*\|. \tag{3.7}$$

Assumption 3.1 also implies the following result [11].

Lemma 3.1. *Let Assumption 3.1 hold. Then*

$$\frac{\|\big(\nabla f(x_{k+1})-\nabla f(x_k)\big)-\nabla^2 f(x^*)s_k\|}{\|s_k\|}\leq M_3\{\|x_{k+1}-x^*\|+\|x_k-x^*\|\} \tag{3.8}$$

holds for some positive constant M_3.

Lemma 3.2. *Let Assumption 3.1 hold and the sequence $\{x_k\}$ be generated by Algorithm 2.1, then there exists a constant $M_6>0$ such that*

$$\frac{\|y_k-\nabla^2 f(x^*)s_k\|}{\|s_k\|}\leq M_6\{\|x_{k+1}-x^*\|+\|x_k-x^*\|\}. \tag{3.9}$$

Proof. From (2.13), we obtain

$$\begin{aligned}
&\bar{g}_{k+1}-\nabla f(x_{k+1})\\
=&\begin{pmatrix}\int_0^1\big(\nabla f(x_{k+1}+t\alpha_{k-1}\|F_{k+1}\|^2e_1)-\nabla f(x_{k+1})\big)^Te_1dt\\ \vdots\\ \int_0^1\big(\nabla f(x_{k+1}+t\alpha_{k-1}\|F_{k+1}\|^2e_n)-\nabla f(x_{k+1})\big)^Te_ndt\end{pmatrix}\\
=&\begin{pmatrix}\int_0^1\big(\int_0^1\nabla^2 f(x_{k+1}+t\tau\alpha_{k-1}\|F_{k+1}\|^2e_1)t\alpha_{k-1}\|F_{k+1}\|^2e_1d\tau\big)^Te_1dt\\ \vdots\\ \int_0^1\big(\int_0^1\nabla^2 f(x_{k+1}+t\tau\alpha_{k-1}\|F_{k+1}\|^2e_n)t\alpha_{k-1}\|F_{k+1}\|^2e_nd\tau\big)^Te_ndt\end{pmatrix}.
\end{aligned}$$

Similarly,

$$g_k-\nabla f(x_k)=\begin{pmatrix}\int_0^1\big(\int_0^1\nabla^2 f(x_k+t\tau\alpha_{k-1}\|F_k\|^2e_1)t\alpha_{k-1}\|F_k\|^2e_1d\tau\big)^Te_1dt\\ \vdots\\ \int_0^1\big(\int_0^1\nabla^2 f(x_k+t\tau\alpha_{k-1}\|F_k\|^2e_n)t\alpha_{k-1}\|F_k\|^2e_nd\tau\big)^Te_ndt\end{pmatrix}.$$

Moreover, Assumption 3.1 implies that there exists a constant $M_4>0$ such that

$$\|\nabla^2 f(x)\|\leq M_4,\quad \forall x\in\Omega_2.$$

Hence, by (2.12), we have

$$\begin{aligned}
& \|y_k - \big(\nabla f(x_{k+1}) - \nabla f(x_k)\big)\| \\
= \;& \|\bar{g}_{k+1} - \nabla f(x_{k+1}) - (g_k - \nabla f(x_k))\| \\
\leq \;& \sum_{i=1}^{n} \Big| \int_0^1 \Big(\int_0^1 \nabla^2 f(x_{k+1} + t\tau\alpha_{k-1}\|F_{k+1}\|^2 e_i) t\alpha_{k-1}\|F_{k+1}\|^2 e_i d\tau\Big)^T e_i dt \\
& - \int_0^1 \Big(\int_0^1 \nabla^2 f(x_k + t\tau\alpha_{k-1}\|F_k\|^2 e_i) t\alpha_{k-1}\|F_k\|^2 e_i d\tau\Big)^T e_i dt \Big| \\
\leq \;& \sum_{i=1}^{n} \Big| \int_0^1 \Big(\int_0^1 \big(\nabla^2 f(x_{k+1} + t\tau\alpha_{k-1}\|F_{k+1}\|^2 e_i) \\
& - \nabla^2 f(x_k + t\tau\alpha_{k-1}\|F_k\|^2 e_i)\big) t\alpha_{k-1}\|F_{k+1}\|^2 e_i d\tau\Big)^T e_i dt \Big| \\
& + \Big| \int_0^1 \Big(\int_0^1 \nabla^2 f(x_k + t\tau\alpha_{k-1}\|F_k\|^2 e_i) t\alpha_{k-1}\big(\|F_k\|^2 - \|F_{k+1}\|^2\big) e_i d\tau\Big)^T e_i dt \Big| \\
\leq \;& \sum_{i=1}^{n} \Big(L_3\big(\|x_{k+1} - x_k\| + \big|\|F_{k+1}\|^2 - \|F_k\|^2\big|\big)\|F_{k+1}\|^2 + M_4\big|\|F_{k+1}\|^2 - \|F_k\|^2\big| \Big) \\
\leq \;& n\Big(L_3\big(\|s_k\| + L_1 m_4(\|x_{k+1} - x^*\| + \|x_k - x^*\|)\|s_k\|\big) m_4^2 \|x_{k+1} - x^*\|^2 \\
& + M_4 L_1 m_4(\|x_{k+1} - x^*\| + \|x_k - x^*\|)\|s_k\| \Big) \\
\leq \;& M_5 \|s_k\| \big(\|x_{k+1} - x^*\| + \|x_k - x^*\|\big)
\end{aligned}$$

with some constant M_5, where the third inequality follows from (3.1) and the fourth inequality uses (3.7) and (2.18). This and (3.8) show that (3.9) holds. □

Remark 3.1 Lemma 3.2 implies that there exists a constant $C_3 > 0$ such that

$$\begin{aligned}
\frac{y_k^T s_k}{\|s_k\|^2} &= \frac{\big(y_k - \nabla^2 f(x^*) s_k\big)^T s_k + s_k^T \nabla^2 f(x^*) s_k}{\|s_k\|^2} \\
&\geq \frac{s_k^T \nabla^2 f(x^*) s_k}{\|s_k\|^2} - \frac{\|y_k - \nabla^2 f(x^*) s_k\|}{\|s_k\|} \geq C_3,
\end{aligned}$$

where we use the facts that $\nabla^2 f(x^*)$ is symmetric positive definite and $x_k \to x^*$. This and $\|F_k\| \to 0$ show that for sufficiently large k, B_{k+1} in (2.11) is always updated by the BFGS formula $B_{k+1} = B_k - \frac{B_k s_k s_k^T B_k}{s_k^T B_k s_k} + \frac{y_k y_k^T}{y_k^T s_k}$. Therefore, without loss of generality, we assume that B_{k+1} is updated by the BFGS formula for all $k > 0$, that is, $A_k = \{1, 2, \cdots, k\}$ and $\hat{K} = \{1, 2, \cdots\}$.

The following lemma gives a bound from below for the stepsize α_k with $k \in K$.

Lemma 3.3. *If $\alpha_k \neq 1$, then there exists a constant $C_2 > 0$ such that*

$$\alpha_k \geq C_2, \quad \forall k \in K.$$

Proof. From the mean value theorem, there exists $t_k \in (0, 1)$ such that

$$\begin{aligned}
& f(x_k + \alpha_k' d_k) - f(x_k) \\
= \;& \nabla f(x_k + t_k \alpha_k' d_k)^T \alpha_k' d_k \\
= \;& \alpha_k' g_k^T d_k + \alpha_k' (\nabla f(x_k) - g_k)^T d_k + \big(\nabla f(x_k + t_k \alpha_k' d_k) - \nabla f(x_k)\big)^T \alpha_k' d_k \\
\leq \;& \alpha_k' g_k^T d_k + n L_2 \alpha_k' \alpha_{k-1} \|F_k\|^2 \|d_k\| + L_2 (\alpha_k')^2 \|d_k\|^2,
\end{aligned}$$

where $\alpha'_k = \frac{\alpha_k}{\rho}$ and the last inequality follows from (2.5) and (2.18). This and (2.14) yield

$$\alpha'_k \geq \frac{-g_k^T d_k - nL_2\alpha_{k-1}\|F_k\|^2\|d_k\|}{(\sigma_1 + L_2)\|d_k\|^2 + \sigma_2\|F_k\|^2}. \tag{3.10}$$

This together with (2.23), (2.6) and (3.6) shows that there exists a constant $C_2 > 0$ such that $\alpha_k \geq C_2, \forall k \in K$. $\square$

The following result shows that Algorithm 2.1 converges linearly.

Theorem 3.4. *Let Assumption 3.1 hold and the sequence $\{x_k\}$ be generated by Algorithm 2.1, then there exist constants $r \in (0,1)$ and $m_5 > 0$ such that*

$$\|F_k\| \leq m_5 r^k, \quad \|x_k - x^*\| \leq m_5 r^k. \tag{3.11}$$

Proof. Let K_1 be the set of index k which satisfies (2.10). Then for $k \in K_1$, we have

$$f(x_{k+1}) \leq \rho_0^2 f(x_k). \tag{3.12}$$

Since $\eta_k \to 0$, we assume $\eta_k \leq \sigma_2 C_2^2$. For $k \in K\backslash K_1$, by (2.9) and Lemma 3.3, we get

$$f(x_{k+1}) \leq (1 - 2\sigma_2\alpha_k^2 + \eta_k) f(x_k) \leq (1 - \sigma_2 C_2^2) f(x_k). \tag{3.13}$$

Let $r_1 = \max\{\sqrt{1 - \sigma_2 C_2^2}, \rho_0\}$, then $r_1 < 1$. By Lemma 2.2 and (3.12)-(3.13), we know that for at least $\lceil k/2 \rceil$ many $i \leq k$ such that

$$\|F_i\| \leq r_1 \|F_{i-1}\|. \tag{3.14}$$

Denote by I_k the set of index $i \leq k$ which satisfies (3.14). By recurrence, (3.14) and the line search (2.9), we obtain that

$$\|F_{k+1}\| \leq \sqrt{\prod_{i\in\{1,2,\cdots,k\}} (1+\eta_i)}\; r_1^{\sum_{i\in I_k}} \|F_0\| \leq r_1^{k/2-1}\sqrt{e^{\eta}}\|F_0\|,$$

where the last inequality uses (2.15). Therefore, $\|F_{k+1}\| \leq \frac{\sqrt{e^{\eta}}\|F_0\|}{r_1 r} r^{k+1}$ with $r = \sqrt{r_1} < 1$. This and (3.7) yield (3.11) with some positive constant m_5. $\square$

Moreover, it follows from (3.11) that

$$\sum_{k=0}^{\infty} \|x_k - x^*\| < \infty. \tag{3.15}$$

(3.15) and (3.9) show that the following Dennis-Moŕe condition holds.

Lemma 3.5 (Theorem 3.2 in [1])**.** *Let Assumption 3.1 hold. Then we have*

$$\lim_{k\to\infty} \frac{\|\big(B_k - \nabla^2 f(x^*)\big)s_k\|}{\|s_k\|} = 0. \tag{3.16}$$

Moreover, the sequences $\{B_k\}$ and $\{B_k^{-1}\}$ are uniformly bounded.

The following result shows that the unit stepsize will be always accepted finally.

Lemma 3.6. *Let Assumption 3.1 hold. Then for all sufficiently large k, we have $\alpha_k = 1$.*

Proof. From (2.5) and Lemma 3.5, there exist two constants M_7 and M_8 such that

$$\|d_k\| = \| - B_k^{-1} g_k\| \leq M_7 \|g_k\| \leq M_7(\|\nabla f(x_k)\| + \|g_k - \nabla f(x_k)\|) \leq M_8 \|F_k\|. \tag{3.17}$$

This together with (3.6) and (3.16) means

$$\begin{aligned}
& \nabla^2 f(x^*)(x_k + d_k - x^*) \\
= \;& \nabla^2 f(x^*)(x_k - x^*) + \nabla^2 f(x^*) d_k \\
= \;& \nabla^2 f(x^*)(x_k - x^*) + B_k d_k + \big(\nabla^2 f(x^*) - B_k\big) d_k \\
= \;& \nabla^2 f(x^*)(x_k - x^*) - (\nabla f(x_k) - \nabla f(x^*)) + (\nabla f(x_k) - g_k) + \big(\nabla^2 f(x^*) - B_k\big) d_k \\
= \;& o(\|x_k - x^*\|) + (\nabla f(x_k) - g_k) + o(\|d_k\|).
\end{aligned}$$

From the above equality, (2.5), (3.17) and (3.7), we know

$$\|\nabla^2 f(x^*)(x_k + d_k - x^*)\| = o(\|x_k - x^*\|) + o(\|F_k\|) = o(\|x_k - x^*\|).$$

Since $\nabla^2 f(x^*)$ is symmetric positive definite, hence the above equality yields

$$\lim_{k\to\infty} \frac{\|x_k + d_k - x^*\|}{\|x_k - x^*\|} = 0. \tag{3.18}$$

Moreover, by (2.18) and (3.7), we obtain

$$\begin{aligned}
\|F(x_k + d_k)\| &= \|F(x_k + d_k) - F(x^*)\| \\
&\leq L_1 \|x_k + d_k - x^*\| \\
&= \frac{L_1}{m_3} \frac{\|x_k + d_k - x^*\|}{\|x_k - x^*\|} m_3 \|x_k - x^*\| \\
&\leq \frac{L_1}{m_3} \frac{\|x_k + d_k - x^*\|}{\|x_k - x^*\|} \|F(x_k)\|,
\end{aligned}$$

which together with (3.18) implies that (2.10) holds for sufficiently large k, that is, $\alpha_k = 1$ for sufficiently large k. □

Lemmas 3.6 and 3.5 imply the superlinear convergence of Algorithm 2.1.

Theorem 3.7 ([11, Theorem 5.4.6]). *Let Assumption 3.1 hold. Then the sequence $\{x_k\}$ be generated by Algorithm 2.1 converges superlinearly, that is, $\alpha_k \equiv 1$ for sufficiently large k and*

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

4 Numerical Experiments

In this section, we first compared the performance of the GN-BFGS method in [6], the cautious BFGS method (CBFGS) proposed by Li and Fukushima in [7] and Algorithm 2.1 for general nonlinear equations (1.1). Then we tested the efficiency of the CBFGS method and Algorithm 2.1 on some nonlinear least squares problems (2.28). The codes were written in Matlab 7.4.

• In the GN-BFGS method, we set the same parameters as those of [6], that is, $B_0 = I$, $r = 0.1$, $\rho = \sqrt{0.9}$, $\sigma_1 = \sigma_2 = 10^{-5}$, $\lambda_{-1} = 0.01$ and $\epsilon_k = k^{-2}$.

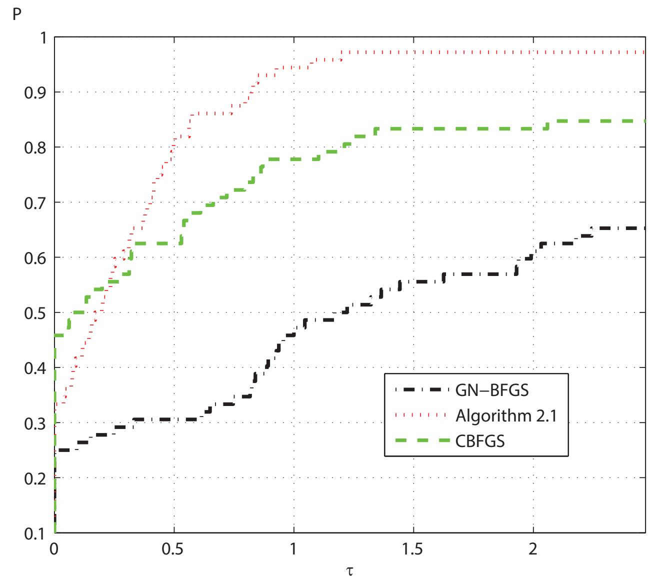

Figure 1: Performance profiles with respect to the number of iterations.

• In Algorithm 2.1, we set parameters $B_0 = I$, $r = 0.1$, $\rho_0 = \sqrt{0.9}$, $\sigma_1 = \sigma_2 = 10^{-5}$, $\alpha_{-1} = 0.01$, $\eta_k = k^{-2}$ and $\mu = 10^{-6}$.

• In the CBFGS method [7], we approximately compute $\nabla f(x_k)$ by the classical standard finite difference $\left(\frac{f(x_k+he_1)-f(x_k)}{h}, \dots, \frac{f(x_k+he_n)-f(x_k)}{h}\right)^T$ with $h = 10^{-15}$ and $h = 10^{-12}$ for the nonlinear equations and the nonlinear least squares problems, respectively. We adopted the standard Armijo line search to compute the stepsize α_k. We set parameters $B_0 = I$, $\rho = 0.1$, $\sigma = 0.1$, $\alpha = 1$ and $\epsilon = 10^{-6}$.

(i) When we tested general nonlinear equations (1.1), we stopped the iteration if the total number of iterations exceeds 200 or $\|F_k\| \le 10^{-5}$. The test problems were created by the following three matrices $A_1, A_2, A_3 \in R^{n\times n}$ and two functions $H : R^n \to R^n$, $W : R^n \to R^n$, where $H(x) = (e^{x_1}-1, e^{x_2}-1, \cdots, e^{x_n}-1)^T$, $W(x) = (\sin x_1 - 1, \sin x_2 - 1, \cdots, \sin x_n - 1)^T$ and

$$A_1 = \begin{pmatrix} 2 & -1 & & & \\ -1 & 2 & -1 & & \\ & \ddots & \ddots & \ddots & \\ & & \ddots & \ddots & -1 \\ & & & -1 & 2 \end{pmatrix}, \quad A_2 = \begin{pmatrix} 2 & -1 & & & \\ 1 & 2 & -1 & & \\ & \ddots & \ddots & \ddots & \\ & & \ddots & \ddots & -1 \\ & & & 1 & 2 \end{pmatrix}, \quad A_3 = \begin{pmatrix} 2 & -1 & & & \\ 0 & 2 & -1 & & \\ & \ddots & \ddots & \ddots & \\ & & \ddots & \ddots & -1 \\ & & & 0 & 2 \end{pmatrix},$$

Then we construct six functions $F(x)$ in the system (1.1) as follows:

Problem1. $F(x) = A_1x + H(x)$;	Problem2. $F(x) = A_1x + W(x)$;
Problem3. $F(x) = A_2x + H(x)$;	Problem4. $F(x) = A_2x + W(x)$;
Problem5. $F(x) = A_3x + H(x)$;	Problem6. $F(x) = A_3x + W(x)$.

It is clear that Problems 1-2 are symmetric and Problems 3-6 are nonsymmetric.

Table 1 lists the numerical results of the three methods on the six problems with different initial points and different n values. In Table 1, "P", "Initial" and "Time" stand for the test

Table 1: Test results for the three methods on nonlinear equations.

			GN-BFGS			Algorithm 2.1			CBFGS		
P	Initial	n	Iter	Time	$\|F_k\|$	Iter	Time	$\|F_k\|$	Iter	Time	$\|F_k\|$
1	x_0	10	11	1.048	2.8215e-007	11	0.90511	2.6128e-006	16	0.92351	6.0859e-006
	x_0	20	17	1.0584	6.7266e-006	24	1.0061	7.0196e-006	28	0.83569	4.3231e-006
	x_0	50	33	0.97135	5.3601e-006	45	1.9508	9.3449e-006	51	2.1059	7.6565e-006
	x_0	100	54	1.5096	8.5778e-006	57	6.0657	7.331e-006	125	12.8936	8.8467e-006
	$\bar{x}_0$	10	10	1.1703	1.481e-007	9	1.1848	1.1755e-006	16	0.86611	7.3687e-006
	$\bar{x}_0$	20	16	0.76968	5.2813e-006	15	1.0146	2.8183e-006	26	0.99583	3.4271e-006
	$\bar{x}_0$	50	34	0.88546	7.9491e-006	27	1.7207	8.6756e-006	31	1.6723	8.5909e-006
	$\bar{x}_0$	100	52	1.2655	7.0907e-006	31	3.6561	9.9592e-006	34	7.4336	7.8293e-006
	$\hat{x}_0$	10	19	1.0596	2.2397e-006	40	1.1377	7.9921e-006	29	1.0659	6.0842e-006
	$\hat{x}_0$	20	27	0.99892	2.7856e-006	48	1.1798	4.0396e-006	39	1.1566	6.9262e-006
	$\hat{x}_0$	50	55	1.5882	4.3485e-006	78	3.1516	4.0693e-006	100	4.0569	9.5394e-006
	$\hat{x}_0$	100	91	1.7503	8.6119e-006	8	2.206	NaN	132	13.0432	8.3809e-006
2	x_0	10	23	1.0447	6.6131e-006	31	1.1018	6.2191e-006	24	0.93773	4.9769e-006
	x_0	20	57	1.2079	2.8553e-006	48	1.1943	6.705e-006	200	2.1172	1.3062
	x_0	50	65	1.1639	9.5702e-006	115	4.1182	4.9732e-006	3	2.0117	NaN
	x_0	100	200	2.5244	1.3393e-005	200	19.6632	2.9976	4	3.1377	NaN
	$\bar{x}_0$	10	21	1.1223	2.7352e-006	35	1.0095	8.1695e-006	23	0.89339	4.1627e-006
	$\bar{x}_0$	20	54	0.86599	8.4346e-006	67	1.2932	4.1925e-006	116	1.7632	9.8718e-006
	$\bar{x}_0$	50	84	0.99273	8.8706e-006	112	3.9176	8.7568e-006	200	8.5619	6.116
	$\bar{x}_0$	100	200	2.171	0.00042728	200	20.8686	3.0456	3	3.1105	NaN
	$\hat{x}_0$	10	20	1.1288	2.5224e-006	36	1.057	7.3856e-006	25	0.9341	4.218e-006
	$\hat{x}_0$	20	47	1.3982	5.4395e-006	62	1.3342	2.8471e-006	59	1.3272	5.3204e-006
	$\hat{x}_0$	50	135	1.0641	7.0401e-006	150	5.136	5.1979e-006	146	7.6011	NaN
	$\hat{x}_0$	100	200	2.1301	0.0022011	200	22.5321	3.0248	9	3.5232	NaN
3	x_0	10	200	1.4679	0.32355	21	0.93261	6.1808e-007	20	0.8854	1.0502e-006
	x_0	20	200	1.3095	0.84298	40	1.1322	1.6205e-006	34	1.2953	9.6677e-006
	x_0	50	200	1.7253	1.8638	52	2.5426	8.6451e-006	37	1.9244	8.0854e-006
	x_0	100	200	2.5966	1.6114	56	6.2791	9.9395e-006	47	5.2077	9.0332e-006
	$\bar{x}_0$	10	200	1.301	0.075087	12	0.85863	9.9741e-006	19	0.93545	2.1489e-006
	$\bar{x}_0$	20	200	1.3559	0.10286	21	0.94522	4.4428e-006	31	1.0888	5.9018e-006
	$\bar{x}_0$	50	200	1.427	0.18978	39	2.0498	7.3718e-006	34	2.1647	7.9365e-006
	$\bar{x}_0$	100	200	2.5813	0.18872	40	4.9906	9.0476e-006	34	4.6117	9.3205e-006
	$\hat{x}_0$	10	200	1.3603	2.7621	6	0.79223	NaN	23	0.98528	2.1419e-006
	$\hat{x}_0$	20	200	1.3822	3.089	51	1.1539	4.6276e-006	39	1.4408	1.9136e-006
	$\hat{x}_0$	50	200	1.4335	3.154	62	2.6105	9.3295e-006	75	3.4425	6.8842e-006
	$\hat{x}_0$	100	200	2.6232	3.146	93	9.8708	8.602e-006	170	19.9981	9.0265e-006

Table 1 continued.

			GN-BFGS			Algorithm 2.1			CBFGS		
P	Initial	n	Iter	Time	$\|F_k\|$	Iter	Time	$\|F_k\|$	Iter	Time	$\|F_k\|$
4	x_0	10	200	1.3843	1.4285	23	0.85771	3.3178e-007	20	0.91863	6.1212e-006
	x_0	20	200	1.4449	2.8213	44	1.1104	2.1014e-006	36	1.1028	6.1736e-006
	x_0	50	200	1.4304	4.5608	103	3.4149	6.9242e-006	45	2.2621	8.9213e-006
	x_0	100	200	2.6736	5.3408	77	8.784	9.9594e-006	3	1.9267	NaN
	$\bar{x}_0$	10	200	1.4819	1.954	25	0.8534	1.6403e-006	20	0.90497	3.4281e-006
	$\bar{x}_0$	20	200	1.135	2.6132	43	1.1326	5.1442e-006	37	1.0996	5.2411e-006
	$\bar{x}_0$	50	200	1.7197	5.5081	71	2.9305	8.5732e-006	60	3.0253	9.2657e-006
	$\bar{x}_0$	100	200	2.6091	6.4258	79	8.7997	8.3433e-006	200	29.8352	3.4998
	$\hat{x}_0$	10	200	1.3764	1.697	21	0.88366	1.8002e-006	19	0.9236	9.5928e-006
	$\hat{x}_0$	20	200	1.4405	1.7151	39	1.1148	7.9653e-006	35	1.3136	2.7938e-006
	$\hat{x}_0$	50	200	1.6869	4.9869	97	3.5427	7.6832e-006	51	2.6815	9.5552e-006
	$\hat{x}_0$	100	200	4.802	9.8699	77	9.8851	9.5321e-006	17	4.9677	NaN
5	x_0	10	86	1.0673	8.8464e-006	19	0.82017	6.0455e-006	20	0.82784	2.5122e-006
	x_0	20	85	0.77264	9.8059e-006	33	0.99799	7.4934e-006	34	1.0207	2.7689e-006
	x_0	50	90	1.1134	1.4171e-006	50	2.5662	8.3434e-006	47	2.2716	8.9916e-006
	x_0	100	145	1.9561	5.5596e-006	51	7.5112	9.2228e-006	47	6.2081	8.4677e-006
	$\bar{x}_0$	10	69	1.0553	9.5372e-006	18	0.84247	3.6371e-006	20	0.8318	1.582e-006
	$\bar{x}_0$	20	72	1.0994	7.4875e-006	33	1.0146	2.7531e-006	32	1.0331	7.3799e-006
	$\bar{x}_0$	50	84	1.1809	4.4746e-006	39	2.1663	8.513e-006	36	2.0324	8.3393e-006
	$\bar{x}_0$	100	151	1.984	8.5627e-006	40	4.8143	9.9909e-006	38	5.0642	9.5722e-006
	$\hat{x}_0$	10	37	1.0757	2.9395e-006	31	0.87309	9.7853e-006	21	0.83738	1.2274e-006
	$\hat{x}_0$	20	85	1.0862	9.1519e-006	47	1.0794	5.0535e-007	34	1.0203	3.1146e-006
	$\hat{x}_0$	50	91	1.2883	6.0859e-006	75	2.969	2.7442e-006	58	2.8185	8.6993e-006
	$\hat{x}_0$	100	154	2.0383	6.6776e-006	87	10.363	8.4248e-006	108	14.9576	9.0055e-006
6	x_0	10	90	1.1004	9.0485e-006	33	0.88687	2.3348e-006	19	0.77249	5.3494e-006
	x_0	20	135	1.0271	9.9744e-006	41	1.0766	5.7777e-006	33	1.0475	6.5278e-006
	x_0	50	102	1.6068	7.4259e-006	76	3.0147	8.3818e-006	57	2.884	9.1108e-006
	x_0	100	154	1.888	5.4206e-006	81	8.1003	8.6528e-006	3	1.823	NaN
	$\bar{x}_0$	10	42	0.93437	8.4974e-006	31	0.8971	2.5814e-006	21	0.8519	8.3409e-006
	$\bar{x}_0$	20	141	1.4067	9.7021e-006	38	1.0065	9.563e-006	37	1.0576	1.6761e-006
	$\bar{x}_0$	50	91	1.0003	9.0507e-006	73	3.0067	9.3243e-006	49	5.4197	7.7265e-006
	$\bar{x}_0$	100	161	2.1892	9.189e-006	78	7.7524	9.4776e-006	3	2.6181	NaN
	$\hat{x}_0$	10	37	1.096	7.9541e-006	26	0.96237	5.4964e-006	20	0.86068	8.4117e-007
	$\hat{x}_0$	20	87	0.90113	9.0594e-006	36	0.98838	4.1e-006	32	1.0068	8.3151e-006
	$\hat{x}_0$	50	124	1.536	9.6887e-006	81	3.128	4.8315e-006	53	3.5589	NaN
	$\hat{x}_0$	100	161	2.0016	8.9602e-006	83	9.0375	8.1393e-006	52	7.39	NaN

Table 2: Test results of Algorithm 2.1 and the CBFGS method on some nonlinear least squares problems.

				Algorithm 2.1			CBFGS		
P	Initial	n	m	Iter	$\|\nabla f(x_k)\|$	$\|F_k\|$	Iter	$\|\nabla f(x_k)\|$	$\|F_k\|$
Bard	1	3	15	147	5.8517e-005	0.090636	19	3.8327e-005	0.090636
	10	3	15	148	5.8441e-005	0.090636	50	8.1793e-006	0.090636
	100	3	15	148	5.8568e-005	0.090636	7	NaN	NaN
Gauss	1	3	15	3	5.4894e-006	0.00010822	3	5.4945e-006	0.00010822
	10	3	15	8	1.1393e-017	0.75115	24	3.3911e-005	0.00012191
	100	3	15	2	0	0.75115	4	3.4467e-005	0.63647
Gulf	1	3	10	2	0	0.19621	47	7.4043e-005	0.0002262
	10	3	10	1	2.3853e-016	1.5717e-016	1	2.3853e-016	1.5717e-016
	100	3	10	1	0	0.19621	1	0	0.19621
Box	1	3	10	70	9.7319e-005	6.849e-005	39	8.4792e-005	6.7384e-005
	10	3	10	35	5.6095e-005	0.002708	24	2.8134e-005	0.2749
	100	3	10	55	NaN	NaN	33	3.5693e-005	0.27493
Kowosb	1	4	11	32	8.2917e-005	0.017536	30	8.3724e-006	0.017536
	10	4	11	188	9.3601e-005	0.029762	44	5.5791e-005	0.029564
	100	4	11	34	8.837e-005	0.036498	44	4.7101e-005	0.04245
Biggs	1	6	13	**313**	2.7977e-005	0.075204	110	1.1731e-005	0.0002578
	10	6	13	65	5.9716e-005	0.075205	105	5.7778e-005	0.00088187
Trig	1	10	10	29	8.3729e-005	0.0052876	30	2.0803e-005	0.0052873
		20	20	51	8.9498e-005	0.0026387	54	9.197e-005	0.0026286
		50	50	40	9.0067e-005	0.0023461	54	9.3835e-005	0.0023438
		100	100	38	8.9446e-005	0.0015715	**500**	**0.00011511**	0.0014006

problem, the initial point and the CPU time in seconds, respectively; "Iter" and $\|F_k\|$ are the total number of iterations and the norm of F_k at the last iteration; and $x_0 = (0.1, \cdots, 0.1)^T$, $\bar{x}_0 = (0.01, \cdots, 0.01)^T$, $\hat{x}_0 = (1, 1/2, \cdots, 1/n)^T$.

From Table 1, for the symmetric Problems 1-2, the GN-BFGS method is the best. However, for nonsymmetric Problems 3-6, Algorithm 2.1 and CBFGS method are more efficient than the GN-BFGS method. Especially, we note that the GN-BFGS method failed to solve Problems 3-4. In our numerical experiments, we observed that Algorithm 2.1 and the CBFGS method need more CPU time for many problems. This is due to the fact that the approximation of g_k requires n F-evaluations, while in the GN-BFGS method such approximation is obtained performing just one F-evaluation.

In order to show the performance of the three methods clearly, we plotted Figure 1 according to the data in Table 1 by using the performance profiles of Dolan and Moré [4]. From Table 1 and Figure 1, we can see that Algorithm 2.1 is the most efficient among the three methods since its performance curves corresponding to the number of iterations is top in the figure.

(ii) When we tested nonlinear least squares problems (2.28), we stopped the iteration if $\|\nabla f(x_k)\| \leq 10^{-4}$ or the total number of iterations exceeds 500. The test problems come from [10]. Table 2 lists the numerical results of Algorithm 2.1 and the CBFGS method with $\nabla f(x_k)$ approximated by the standard finite difference approach. The second column of Table 2 means that the initial point is $x_0, 10x_0, 100x_0$, where x_0 is suggested by Moré et al. in [10]. From Table 2, we see that both methods performed well. For some problems such as "Bard", the CBFGS method is better. One possible reason is that Algorithm 2.1 may not converge superlinearly since g_k is no longer a good approximation of $\nabla f(x_k)$ in the nonzero residual case (otherwise, $\alpha_k = 1$ and $\|F(x_k)\| \to \|F(x^*)\| \neq 0$, which implies that $\alpha_{k-1}\|F(x_k)\|^2$ can not converges to 0). On the other hand, for the problem "Trig" with $m = n = 100$, we see that Algorithm 2.1 performs better than the CBFGS method. This shows that Algorithm 2.1 converges globally. However, the CBFGS method with such standard finite difference approximation does not converge for this problem.

5 Conclusions

In this paper, we proposed a globally and superlinearly convergent BFGS method for general nonlinear equations without using exact gradient and Jacobian. This extended the GN-BFGS method proposed by Li and Fukushima [6]. Some numerical results compared with the GN-BFGS method and the CBFGS method are also reported.

Acknowledgments

We would like to thank the editor and the referees whose very helpful suggestions led to much improvement of this paper.

References

[1] R. H. Byrd and J. Nocedal, A tool for the analysis of quasi-Newton methods with application to unconstrained minimization, *SIAM J. Numer. Anal.* 26 (1989) 727–739.

[2] H. Cao and D. Li, Adjoint Broyden methods for symmetric nonlinear equations, *Pacific J. Optim.* 13 (2017) 645–663.

[3] J. E. Dennis and J. J. Moré, A characterization of superlinear convergence and its applications to quasi-Newton methods, *Math. Comp.* 28 (1974) 549–560.

[4] E. D. Dolan and J. J. Moré, Benchmarking optimization software with performance profiles, *Math. Program.* 91 (2002) 201–213.

[5] D. Li and W. Cheng, Recent progress in the global convergence of quasi-Newton methods for nonlinear equations, *Hokkaido Math. J.* 36 (2007) 729–743.

[6] D. Li and M. Fukushima, A globally and superlinearly convergent Gauss-Newton-based BFGS method for symmetric nonlinear equations, *SIAM J. Numer. Anal.* 37 (2000) 152–172.

[7] D. Li and M. Fukushima, On the global convergence of the BFGS method for nonconvex unconstrained optimization problems, *SIAM J. Optim.* 11 (2001) 1054–1064.

[8] Q. Li and D. Li, A class of derivative-free methods for large-scale nonlinear monotone equations, *IMA J. Numer. Anal.* 31 (2011) 1625–1635.

[9] J. M. Martínez, Practical quasi-Newton methods for solving nonlinear systems, *J. Comput. Appl. Math.* 124 (2000) 97–121.

[10] J. J. Moré, B. S. Garbow and K. H. Hillstrom, Testing unconstrained optimization software, *ACM Trans. Math. Softw.* 7 (1981) 17–41.

[11] W. Sun and Y. Yuan, *Optimization Theory and Methods*, Springer Science and Business Media, LLC, New York, 2006.

[12] Z. Wang and Y. Chen, S. Huang and D. Feng, A modified nonmonotone BFGS algorithm for solving smooth nonlinear equations, *Optim. Lett.* 8 (2014) 1845–1860.

[13] G. Yuan and X. Lu, A new backtracking inexact BFGS method for symmetric nonlinear equations, *Comput. Math. Appl.* 55 (2008) 116–129.

[14] G. Yuan, Z. Sheng, B. Wang, W. Hu and C. Li, The global convergence of a modified BFGS method for nonconvex functions, *J. Comput. Appl. Math.* 327 (2018) 274–294.

[15] G. Yuan, Z. Wei and X. Lu, Global convergence of BFGS and PRP methods under a modified weak Wolfe-Powell line search, *Appl. Math. Model.* 47 (2017) 811–825.

[16] G. Yuan, Z. Wei and X. Lu, A BFGS trust-region method for nonlinear equations, *Computing* 92 (2011) 317–333.

[17] G. Yuan and S. Yao, A BFGS algorithm for solving symmetric nonlinear equations, *Optimization* 62 (2013) 45–64.

[18] Y. Yuan, Recent advances in numerical methods for nonlinear equations and nonlinear least squares, *Numer. Algebra Control Optim.* 1 (2011) 15–34.

[19] L. Zhang, A derivative-free conjugate residual method using secant condition for general large-scale nonlinear equations, *Numer. Algor.* 83 (2020)1277–1293.

[20] L. Zhang and H. Tang, A hybrid MBFGS and CBFGS method for nonconvex minimization with a global complexity bound, *Pacific J. Optim.* 14 (2018) 693–702.

[21] W. Zhou, A Gauss-Newton-based BFGS method for symmetric nonlinear least squares problems, *Pacific J. Optim.* 9 (2013) 373–389.

[22] W. Zhou, A modified BFGS type quasi-Newton method with line search for symmetric nonlinear equations problems, J. Comput. Appl. Math., 367 (2019): 112454.

[23] W. Zhou and X. Chen, Global convergence of a new hybrid Gauss-Newton structured BFGS methods for nonlinear least squares problems, *SIAM J. Optim.* 20 (2010) 2422–2441.

[24] W. Zhou and D. Li, A globally convergent BFGS method for nonlinear monotone equations without any merit functions, *Math. Comp.* 77 (2008) 2231–2240.

[25] W. Zhou and D. Li, On the Q-linear convergence rate of a class of methods for monotone nonlinear equations, *Pacific J. Optim.* 14 (2018) 723–737.

Manuscript received 20 January 2019
revised 30 July 2019
accepted for publication 1 August 2019

Weijun Zhou
Department of Mathematics and Statistics
Changsha University of Science and Technology
Changsha 410114, China
E-mail address: mathwjzhou@csust.edu.cn

Li Zhang
Department of Mathematics and Statistics
Changsha University of Science and Technology
Changsha 410114, China
E-mail address: mathlizhang@126.com