Available online at www.isr-publications.com/jnsa
J. Nonlinear Sci. Appl., 11 (2018), 252–262
Research Article

ISSN: 2008-1898

Journal of Nonlinear Sciences and Applications

Journal Homepage: www.isr-publications.com/jnsa

Two new Newton-type methods for the nonlinear equations

Ya-Jun Xie[a,b], Na Huang[a], Chang-Feng Ma[a,*]

[a] *College of Mathematics and Informatics, Fujian Key Laborotary of Mathematical Analysis and Applications, Fujian Normal University, Fuzhou, 350117, P. R. China.*

[b] *Department of Mathematics and Physics, Fujian Jiangxia University, Fuzhou 350108, P. R. China.*

Communicated by X.-J. Yang

Abstract

In this paper, based on the classical Newton method and Halley method, we propose two new Newton methods for solving the systems of nonlinear equations. The convergence performances of the two new variants of Newton iteration method are analyzed in details. Some numerical experiments are also presented to demonstrate the feasibility and efficiency of the proposed methods.

Keywords: Systems of nonlinear equations, Newton iteration method, Armijo linear search, convergence analysis, numerical tests.

2010 MSC: 65H10.

1. Introduction

In this paper, we discuss the following system of nonlinear equations, to find a vector x such that

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

where $x = (x_1, x_2, \cdots, x_n)^T \in \mathbb{R}^n$, and $f = (f_1, f_2, \cdots, f_n)^T : S \subseteq \mathbb{R}^n \to \mathbb{R}^n$ is continuously differentiable function.

The nonlinear system of form (1.1) has been investigated extensively owing to various scientific and engineering applications [1, 3, 7, 10, 23, 24]. Normally, it can't get the exactly solutions even when n is very small, which promotes greatly the substantial developments of constructing various kinds of iterative methods. Many research works have been done in some literatures on fast solvers for the system of nonlinear equations (1.1). One of the most effective methods is the classical Newton method:

$$x^{k+1} = x^k - (f'(x^k))^{-1} f(x^k), \quad k = 1, 2, \ldots, \tag{1.2}$$

*Corresponding author
Email address: macf@fjnu.edu.cn (Chang-Feng Ma)

doi: 10.22436/jnsa.011.02.07

Received: 2017-01-05 *Revised:* 2017-12-15 *Accepted:* 2017-12-18

where $f'(x^k)$ is the Jacobian matrix of f at the k-th step x^k. The local convergence and quadratic convergence rate of the Newton method attracts many researchers to apply the Newton method (1.2) to other types of problems, such as weakly nonlinear systems [2, 4], complementarity problems [8, 14], and so on. Recently, higher order iterative methods also arouse wide concern. In [11], Levin proposed a directional Halley method with a cubic convergence rate. Then, to avoid computing the second derivative in Halley method, a directional quasi-Halley method with one more function operation per iteration than the directional Newton method was also studied in [11]. By using the decomposition technique, Shah in [18] investigated some higher order iterative schemes for nonlinear equations. A new two-parameter Chebyshev-Halley-like family of fourth and sixth-order approaches was proposed in [13] for the systems of nonlinear equations.

However, when the root x^* of the system (1.1) is multiple, the Newton method (1.2) will be invalid as the Jacobian matrix $f'(x^k)$ is singular or nearly singular if k is sufficiently large. A family of multi-point iterative methods was introduced in [15] for deriving the multiple root of the nonlinear equations (1.1).

In [20], by a transformation $g(x) = \frac{f(x)}{f'(x)}$ instead of $f(x)$ for calculating a multiple root of $f(x) = 0$, Traub utilized the classical Newton method to solve the transformation equation $g(x) = 0$. The concrete iterative form can be written as follows:

$$x^{k+1} = x^k - \left(f'(x^k)(f'(x^k))^T - f(x^k)\nabla^2 f(x^k)\right)^{-1}\left(f(x^k)f'(x^k)\right).$$

Inspired by above work, we establish a new iterative scheme based on the Newton method (1.2) for solving the systems of nonlinear equations (1.1). The new iterative method can be formulated as:

$$x^{k+1} = x^k - \left(\beta f'(x^k)(f'(x^k))^T - \gamma f(x^k)\nabla^2 f(x^k)\right)^{-1}\left(\alpha f(x^k)f'(x^k)\right). \tag{1.3}$$

It is easy to see that the proposed method (1.3) can be reduced to the Halley method [11, 16, 17, 22] if $\alpha = 2$, $\beta = 2$ and $\gamma = -1$, and the classical Newton method if $\alpha = 1$, $\beta = 0$ and $\gamma = -1$. For accelerating the convergence of the scheme (1.3), we also analyse a new method by introducing the Armijo line search technique. For the two above new methods, we establish the convergence under some proper conditions.

The remainder of this paper is organized as follows. In Section 2, two new variants of Newton iteration method are proposed for solving the systems of nonlinear equations (1.1). A detailed discussion on the convergence performances of two new variants is shown in Section 3. In Section 4, a variety of numerical tests are provided to illustrate the superiority of the presented variant 2 of Newton iteration method. Finally, some concluding remarks are given in Section 5.

2. Two new variants of Newton method

In this section, we give two new variants of the classical Newton method and well-known Halley method. Moreover, some necessary assumptions and valuable conclusions are provided, these results contribute significantly to the analysis of the convergence performance of these new variants of Newton method.

As a matter of convenience, we use the following notations throughout this paper: let $x \in \mathbb{R}^n$, $A \in \mathbb{R}^{n\times m}$, $F(x) \in \mathbb{R}$. A^T and A^{-1} denote transpose and inverse of matrix A, respectively. ∇F and $\nabla^2 F$ denote the gradient and Hessian matrix of the differentiable scalar function F, respectively. F_k, ∇F_k, $\nabla^2 F_k$ denote the function values of $F(x^k)$, $\nabla F(x^k)$ and $\nabla^2 F(x^k)$ in the k-th step iteration x^k, respectively. $\|\cdot\|$ denotes the Euclidean norm.

Obviously, the system of nonlinear equations (1.1) can be equivalent to the optimization problem

$$\min_{x\in S} F(x) := \|f(x)\|^2 = \Sigma_{i=1}^n f_i^2(x). \tag{2.1}$$

In the following content, we will introduce the algorithms for solving the problem (2.1).

Assumption 2.1. Let $F(x)$ be twice continuously differential. Assume that the level set defined as:

$$\mathbb{L}(x_0) = \{x : x \in S \mid F(x) \leqslant F(x_0)\}$$

is bounded.

Assumption 2.2. $\nabla^2 F(x)$ is Lipschitz continuous, namely, there exists positive constant L_1 such that

$$\|\nabla^2 F(x) - \nabla^2 F(y)\| \leqslant L_1 \|x - y\|.$$

In fact, $\nabla F(x)$ is also Lipschitz continuous due to its differentiability, hence we immediately have the following inequality

$$\|\nabla F(x) - \nabla F(y)\| \leqslant L_2 \|x - y\|,$$

where L_2 is Lipschitz constant.

Algorithm 2.3 (The variant 1 of newton iteration method (VNM1))**.**

Step 1. Give the initial guess x^0, the parameters $\alpha,\ \beta,\ \gamma$. Let $\mu > 0$ and any small positive number ε. Set $k := 0$.

Step 2. If $\|\nabla F_k\| < \varepsilon$, stop.

Step 3. Solve the linear system

$$(\beta \nabla F_k \nabla F_k^T - \gamma F_k \nabla^2 F_k) p^k = -\alpha F_k \nabla F_k.$$

If the coefficient matrix $\beta \nabla F_k \nabla F_k^T - \gamma F_k \nabla^2 F_k$ is singular, then solve

$$(\beta \nabla F_k \nabla F_k^T - \gamma F_k \nabla F_k^2 + \mu I) p^k = -\alpha F_k \nabla F_k.$$

Step 4. Update the iterative sequence

$$x^{k+1} = x^k + p^k.$$

Set $k := k + 1$, return to Step 2.

It is easy to see that the step-size in Algorithm 2.3 is identically equal to 1. Once the initial guess was not chosen well, the convergence of Algorithm 2.3 can not be guaranteed. So we introduce the Armijo line search technique and give the following algorithm.

Algorithm 2.4 (The variant 2 of Newton iteration method (VNM2))**.**

Step 1. Give the initial guess x^0, the parameters $\sigma \in (0, 0.5),\ \rho \in (0, 1),\ \beta,\ \gamma$. Let $\mu > 0$ and any small positive number ε. Set $k := 0$.

Step 2. If $\|\nabla F_k\| < \varepsilon$, stop.

Step 3. Solve the linear system

$$(\beta \nabla F_k \nabla F_k^T - \gamma F_k \nabla^2 F_k) p^k = -F_k \nabla F_k.$$

If the coefficient matrix $\beta \nabla F_k \nabla F_k^T - \gamma F_k \nabla^2 F_k$ is singular, then solve

$$(\beta \nabla F_k \nabla F_k^T - \gamma F_k \nabla F_k^2 + \mu I) p^k = -F_k \nabla F_k.$$

Step 4. Find the minimum non-negative integer m such that

$$F(x^k + \rho^m p^k) \leqslant F(x^k) + \sigma \rho^m \nabla F_k^T p^k. \tag{2.2}$$

Let $m_k := m$.

Step 5. Update the iterative sequence

$$x^{k+1} = x^k + \rho^{m_k} p^k.$$

Set $k := k + 1$, return to Step 2.

3. The analysis of convergence

Now we give the following lemmas. First, the famous Sherman-Morrison-Woodbury formula [21] and its variant [9] are exhibited.

Lemma 3.1. *Assume that* $U,\ V \in \mathbb{R}^{n\times m}$ *and* $W \in \mathbb{R}^{n\times n}$ *is a nonsingular matrix. If* $I + VW^{-1}U \in \mathbb{R}^{n\times n}$ *is a nonsingular matrix, then* $W + UV$ *is also nonsingular. Moreover, it satisfies*

$$(W + UV^T)^{-1} = W^{-1} - W^{-1}U(I + V^TW^{-1}U)^{-1}V^TW^{-1},$$

where I *denotes the identity matrix with proper dimension.*

Lemma 3.2. *Assume that* $u,\ v \in \mathbb{R}^n$ *and* $W \in \mathbb{R}^{n\times n}$ *is a nonsingular matrix. If* $1 + v^TW^{-1}u$ *is a non-zero scalar, then* $W + uv^T$ *is nonsingular. Moreover, it holds*

$$(W + uv^T)^{-1} = W^{-1} - \frac{1}{1 + v^TW^{-1}u}W^{-1}uv^TW^{-1},$$

where I *denotes the identity matrix with proper dimension.*

Lemma 3.3. *Suppose that Assumptions 2.1-2.2 hold. If the parameters* β *and* γ *satisfy*

$$\frac{\beta}{\gamma}(\nabla F_k^T(F_k\nabla^2F_k)^{-1}\nabla F_k) \leqslant \frac{1}{2}, \tag{3.1}$$

then

(a) $\left|\dfrac{\beta\nabla F_k^T(\gamma F_k\nabla^2F_k)^{-1}\nabla F_k}{\beta\nabla F_k^T(\gamma F_k\nabla^2F_k)^{-1}\nabla F_k - 1}\right| < 1;$

(b) $(\frac{-\gamma F_k}{\beta}\nabla^2F_k + \nabla F_k\nabla F_k^T)^{-1} = \frac{-\beta}{\gamma F_k}(\nabla^2F_k)^{-1}\Big(I - \frac{\nabla F_k\nabla F_k^T(\frac{-\gamma F_k}{\beta}\nabla^2F_k)^{-1}}{1+\nabla F_k^T(\frac{-\gamma F_k}{\beta}\nabla^2F_k)^{-1}\nabla F_k}\Big).$

Proof. It follows from (3.1) that

$$\beta(\nabla F_k^T(-\gamma F_k\nabla^2F_k)^{-1}\nabla F_k) \geqslant -\frac{1}{2}, \tag{3.2}$$

which leads to the first result immediately.

Let $W = \frac{-\gamma F_k}{\beta}\nabla^2F_k$, $u = v = \nabla F_k$. From Lemmas 3.1-3.2 and (3.2), it yields

$$1 + v^TW^{-1}u = 1 + \beta(\nabla F_k^T(-\gamma F_k\nabla^2F_k)^{-1}\nabla F_k) \geqslant \frac{1}{2} \neq 0. \tag{3.3}$$

This implies that the matrix $W + uv^T$ is invertible. Then we have

$$\begin{aligned}(W + uv^T)^{-1} &= (\frac{-\gamma F_k}{\beta}\nabla^2F_k + \nabla F_k\nabla F_k^T)^{-1}\\ &= (\frac{-\gamma F_k}{\beta}\nabla^2F_k)^{-1} - (\frac{-\gamma F_k}{\beta}\nabla^2F_k)^{-1}\\ &\quad\cdot \nabla F_k\Big(1 + \nabla F_k^T(\frac{-\gamma F_k}{\beta}\nabla^2F_k)^{-1}\nabla F_k\Big)^{-1}\nabla F_k^T(\frac{-\gamma F_k}{\beta}\nabla^2F_k)^{-1}\\ &= \frac{-\beta}{\gamma F_k}(\nabla^2F_k)^{-1}\Big(I - \frac{\nabla F_k\nabla F_k^T(\frac{-\gamma F_k}{\beta}\nabla^2F_k)^{-1}}{1 + \nabla F_k^T(\frac{-\gamma F_k}{\beta}\nabla^2F_k)^{-1}\nabla F_k}\Big).\end{aligned}$$

This completes the proof. □

Theorem 3.4. *Suppose that Assumptions 2.1-2.2 hold. Assume that p^k is generated by Algorithm 2.3, i.e.,*

$$p^k = \alpha(\beta \nabla F_k \nabla F_k^T - \gamma \nabla F_k \nabla^2 \nabla F_k)^{-1}(-F_k \nabla F_k). \tag{3.4}$$

Let

$$\xi_1 := \frac{\delta}{2} + \delta|\alpha + \gamma| + \delta|\alpha| < 1,$$

where $\delta := \frac{\rho L}{\gamma}$, $L := \max\{L_1, L_2\}$, $\rho := \|(\nabla^2 F_0)^{-1}\|$. Then the iterative

$$x^{k+1} = x^k + p^k, \ k = 0, 1, \dots,$$

converges to the solution x^* of the system of nonlinear equations (1.1) if the initial guess is close to the solution x^* sufficiently.

Proof. By (3.4) and Lemma 3.3 (b), we obtain

$$p^0 = \frac{\alpha}{\beta}(\nabla F_0 \nabla F_0^T - \frac{\gamma \nabla F_0}{\beta} \nabla^2 F_0)^{-1}(-F_0 \nabla F_0) = \frac{\alpha}{\gamma}(\nabla^2 F_0)^{-1}\Big(I - \frac{\nabla F_0 \nabla F_0^T (\frac{-\gamma F_0}{\beta} \nabla^2 F_0)^{-1}}{1 + \nabla F_0^T (\frac{-\gamma F_0}{\beta} \nabla^2 F_0)^{-1} \nabla F_0}\Big)\nabla F_0.$$

Using Lemma 3.3 (a), Assumptions 2.1-2.2, and the boundedness of $\|(\nabla^2 F_0)^{-1}\|$, it is not difficult to verify that

$$\|p^0\| \leqslant |\frac{\alpha}{\gamma}| \|(\nabla^2 F_0)^{-1}\| \Big\| \nabla F_0 - \nabla F_0 \frac{\nabla F_0^T (\frac{-\gamma F_0}{\beta} \nabla^2 F_0)^{-1} \nabla F_0}{1 + \nabla F_0^T (\frac{-\gamma F_k}{\beta} \nabla^2 F_0)^{-1} \nabla F_k} \Big\| \leqslant |\frac{2\alpha}{\gamma}| \rho \|\nabla F_0\| \leqslant |\frac{2\alpha}{\gamma}| \rho L \|x^0 - x^*\|,$$

where $\rho := \|(\nabla^2 F_0)^{-1}\|$. Then we can derive

$$\begin{aligned} x^1 - x^* = x^0 + p^0 - x^* &= x^0 - x^* + \frac{\alpha}{\gamma}(\nabla^2 F_0)^{-1}\Big(I - \frac{\nabla F_0 \nabla F_0^T (\frac{-\gamma F_0}{\beta} \nabla^2 F_0)^{-1}}{1 + \nabla F_0^T (\frac{-\gamma F_0}{\beta} \nabla^2 F_0)^{-1} \nabla F_0}\Big)\nabla F_0 \\ &= (\nabla^2 F_0)^{-1}\Big(\nabla^2 F_0 (x^k - x^*) + \frac{\alpha}{\gamma}\Big(I - \frac{\nabla F_0 \nabla F_0^T (\frac{-\gamma F_0}{\beta} \nabla^2 F_0)^{-1}}{1 + \nabla F_0^T (\frac{-\gamma F_0}{\beta} \nabla^2 F_0)^{-1} \nabla F_0}\Big)\nabla F_0\Big) \\ &= (\nabla^2 F_0)^{-1}\Big(\nabla^2 F_0 (x^0 - x^*) + \frac{\alpha}{\gamma} \nabla F_0 - \frac{\alpha}{\gamma} \frac{\nabla F_0 \nabla F_0^T (\frac{-\gamma F_0}{\beta} \nabla^2 F_0)^{-1} \nabla F_0}{1 + \nabla F_0^T (\frac{-\gamma F_0}{\beta} \nabla^2 F_0)^{-1} \nabla F_0}\Big). \end{aligned}$$

Noticing that $\nabla F^* = \nabla F(x^*) = 0$, it follows

$$\begin{aligned} \|x^1 - x^*\| &= \frac{1}{\gamma}\Big\|(\nabla^2 F_0)^{-1}\Big(\gamma \nabla^2 F_0 (x^0 - x^*) + \alpha \nabla F_0 - \alpha \frac{\nabla F_0 \nabla F_0^T (\frac{-\gamma F_0}{\beta} \nabla^2 F_0)^{-1} \nabla F_0}{1 + \nabla F_0^T (\frac{-\gamma F_0}{\beta} \nabla^2 F_0)^{-1} \nabla F_0}\Big)\Big\| \\ &\leqslant \frac{1}{\gamma}\|(\nabla^2 F_0)^{-1}\| \Big\|\gamma \nabla^2 F_0 (x^0 - x^*) + \alpha \nabla F_0 - \alpha \frac{\nabla F_0 \nabla F_0^T (\frac{-\gamma F_0}{\beta} \nabla^2 F_0)^{-1} \nabla F_0}{1 + \nabla F_0^T (\frac{-\gamma F_0}{\beta} \nabla^2 F_0)^{-1} \nabla F_0}\Big\| \\ &\leqslant \frac{1}{\gamma}\|(\nabla^2 F_0)^{-1}\| \Big\|\gamma \nabla^2 F_0 (x^0 - x^*) - \gamma(\nabla F_0 - \nabla F^*) + (\alpha + \gamma)(\nabla F_0 - \nabla F^*) \\ &\quad - \alpha \frac{\nabla F_0 \nabla F_0^T (\frac{-\gamma F_0}{\beta} \nabla^2 F_0)^{-1} \nabla F_0}{1 + \nabla F_0^T (\frac{-\gamma F_0}{\beta} \nabla^2 F_k)^{-1} \nabla F_0}\Big\| \\ &\leqslant \frac{1}{\gamma}\|(\nabla^2 F_0)^{-1}\| \Big[\|\gamma \nabla^2 F_k (x^0 - x^*) - \gamma(\nabla F_0 - \nabla F^*)\| + |\alpha + \gamma| \|\nabla F_0 - \nabla F^*\| \\ &\quad + |\alpha| \|\nabla F_0\| \Big| \frac{\beta \nabla F_0^T (\gamma F_0 \nabla^2 F_0)^{-1} \nabla F_0}{\beta \nabla F_0^T (\gamma F_0 \nabla^2 F_0)^{-1} \nabla F_0 - 1} \Big| \Big]. \end{aligned} \tag{3.5}$$

By the integral mean value theorem, we have

$$\nabla F_0 - \nabla F^* = -\int_0^1 \nabla^2 F(x^0 + \tau(x^* - x^0))(x^* - x^0)d\tau.$$

This shows that

$$\begin{aligned}\|\nabla^2 F_0(x^0 - x^*) - (\nabla F_0 - \nabla F^*)\| &= \left\|\int_0^1 \Big(\nabla^2 F_0 - \nabla^2 F\big(x^0 + \tau(x^* - x^0)\big)\Big)(x^0 - x^*)d\tau\right\| \\ &\leqslant \int_0^1 \|(\nabla^2 F_0 - \nabla^2 F(x^0 + \tau(x^* - x^0))\|\|x^0 - x^*\|d\tau \leqslant \frac{1}{2}L\|x^0 - x^*\|^2.\end{aligned}$$

If the initial guess is close to the x^*, i.e., $\|x^0 - x^*\| \to 0$, by using the fact $\|\nabla F_0\| \leqslant L\|x^0 - x^*\|$, Lemma 3.3 (a), (3.5), and the above inequality, we can get

$$\begin{aligned}\|x^1 - x^*\| &\leqslant \frac{1}{\gamma}\|(\nabla^2 F_0)^{-1}\|\big(\frac{1}{2}L\|x^0 - x^*\|^2 + |\alpha + \gamma|L\|x^0 - x^*\| + |\alpha|L\|x^0 - x^*\|\big) \\ &\leqslant \frac{1}{\gamma}\rho L\big(\frac{1}{2}\|x^0 - x^*\| + |\alpha + \gamma| + |\alpha|\big)\|x^0 - x^*\| \\ &= \delta\big(\frac{1}{2}\|x^0 - x^*\| + |\alpha + \gamma| + |\alpha|\big)\|x^0 - x^*\| < 1,\end{aligned}$$

where $\delta = \frac{1}{\gamma}\rho L$.

By method of induction, we immediately have $\|x^k - x^*\| < 1$ and

$$\begin{aligned}\|x^{k+1} - x^*\| &\leqslant \delta\big(\frac{1}{2}\|x^k - x^*\| + |\alpha + \gamma| + |\alpha|\big)\|x^k - x^*\| \\ &\leqslant \delta\big(\frac{1}{2} + |\alpha + \gamma| + |\alpha|\big)\|x^k - x^*\| \\ &\leqslant \big(\frac{\delta}{2} + \delta|\alpha + \gamma| + \delta|\alpha|\big)^{k+1}\|x^0 - x^*\| = \xi_1^{k+1}\|x^0 - x^*\|,\end{aligned}$$

where $\xi_1 := \frac{\delta}{2} + \delta|\alpha + \gamma| + \delta|\alpha| < 1$. Hence, we have the sequence of $\{x^k\}$ converges to the solution x^* when $k \to \infty$. This completes the proof. □

Next, we will give the important convergence results for the proposed variant 2 of Newton method.

Theorem 3.5. *Assume that $\{x^k\}$ is generated by Algorithm 2.4. For any $x^0 \in \mathbb{R}^n$, the gradient $\nabla F(x)$ is uniformly continuous on the level set*

$$\mathbb{L}(x_0) = \{x : x \in \mathbb{R}^n \mid F(x) \leqslant F(x^0)\}.$$

If $Q := \beta\nabla F_k \nabla F_k^T - \gamma F_k \nabla^2 F_k$ is positive definite. Then it satisfies $\nabla F(x^*) = 0$, i.e., x^* is the stationary point of function $F(x)$, where x^* denotes any accumulation point of the sequence $\{x^k\}$.

Proof. We give the proof by contradiction. Suppose that x^* is the accumulation point of the sequence $\{x^k\}$ and $\nabla F(x^*) \neq 0$. Since the sequence $\{x^k\}$ is bounded on the level set by assumption. Then there exists a convergent subsequence. Without loss of generality, it is still denoted by $\{x^k\}$. Hence $x^k \to x^*$, $F(x^k) \to F(x^*)$, and $F(x^k) - F(x^{k+1}) \to 0$. Furthermore, from the Armijo line search criterion in Algorithm 2.4, we have

$$-\sigma\nabla F_k^T s^k < F(x^k) - F(x^{k+1}) \to 0, \quad \nabla F_k^T s^k \to 0, \tag{3.6}$$

where $s^k := \rho^{m_k} p^k$.

If $\nabla F_k^T \nrightarrow 0$, then by (3.6), we get $\|s^k\| \to 0$. Since m_k is the minimal nonnegative integer such that the inequality (2.2) holding in Algorithm 2.4 .

So, for $\rho^{m_k-1} = \frac{\rho^{m_k}}{\rho}$, the inequality can be written as

$$F(x^k + \rho^{m_k-1}p^k) - F(x^k) > \sigma\rho^{m_k-1}\nabla F_k^T p^k. \tag{3.7}$$

Noticing that $\rho^{m_k-1}p^k = \frac{s^k}{\rho}$ and (3.7), it follows

$$F(x^k + \frac{s^k}{\rho}) - F(x^k) > \sigma\nabla F_k^T(\frac{s^k}{\rho}). \tag{3.8}$$

Set $d^k = \frac{s^k}{\|s^k\|}$. Then $\frac{s^k}{\rho} = \frac{\|s^k\|}{\rho}d^k$. By (3.8) and observing that $\|s^k\| \to 0$, we get

$$\alpha_k' := \frac{\|s^k\|}{\rho} \to 0$$

and

$$\frac{F(x^k + \alpha_k' d^k) - F(x^k)}{\alpha_k'} > \sigma\nabla F_k^T d^k. \tag{3.9}$$

Noticing that $\|d^k\| = 1$, we can know that the sequence $\{\|d^k\|\}$ is bounded. Then there exists a convergent subsequence. Without loss of generality, we still denote it by $\|d^k\|(\to \|d^*\|)$, where $\|d^*\| = 1$. Executing the limit operation on the both sides of (3.9), we have

$$\nabla F(x^*)^T d^* \geqslant \sigma\nabla F(x^*)^T d^*.$$

By $\sigma < 1$, we obtain

$$\nabla F(x^*)^T d^* \geqslant 0. \tag{3.10}$$

On the other hand, observing that $d^k = \frac{s^k}{\|s^k\|} = \frac{p^k}{\|p^k\|}$. If $Q := \beta\nabla F_k \nabla F_k^T - \gamma F_k \nabla^2 F_k$ is positive definite, it yields that

$$-\nabla F_k^T d^k = -\nabla F_k^T \frac{p^k}{\|p^k\|} = \frac{(p^k)^T(\beta\nabla F_k\nabla F_k^T - \gamma F_k\nabla^2 F_k)^T p^k}{F_k\|p^k\|} = \frac{(p^k)^T Q^T p^k}{F_k\|p^k\|} > 0. \tag{3.11}$$

If P is non-positive definite, by Algorithm 2.4, we can choose the proper parameter $\tau > 0$ such that $\widehat{Q} := \beta\nabla F_k \nabla F_k^T - \gamma F_k \nabla^2 F_k + \tau I$ is a positive definite matrix. Then it gives that

$$-\nabla F_k^T d^k = -\nabla F_k^T \frac{p^k}{\|p^k\|} = \frac{(p^k)^T(\beta\nabla F_k\nabla F_k^T - \gamma F_k\nabla^2 F_k + \tau I)^T p^k}{F_k\|p^k\|} = \frac{(p^k)^T(\widehat{Q})^T p^k}{F_k\|p^k\|} > 0. \tag{3.12}$$

It follows from (3.11) or (3.12) that

$$\nabla F(x^*)^T d^* < 0,$$

which is contradictory with (3.10). This completes the proof. □

Theorem 3.6. *Assume that the conditions of Theorem 3.5 hold. For any $p \in \mathbb{R}^n$, $x \in \mathbb{L}(x_0)$, if $Q := \beta\nabla F_k \nabla F_k^T - \gamma F_k \nabla^2 F_k$ is positive definite and*

$$p^T Q p \geqslant \sigma_1\|p\|^2. \tag{3.13}$$

Otherwise if Q is non-positive definite, but $Q + \tau I$ is positive definite and

$$p^T(Q + \tau I)p \geqslant \sigma_2\|p\|^2, \tag{3.14}$$

where σ_1, σ_2, $\tau > 0$, then the stationary point x^* of function $F(x)$ is also the global minimum point, i.e, $F(x^*) = 0$.

Proof. It follows from (3.13) or (3.14) that the function $F(x)$ is uniformly convex on the level set $\mathbb{L}(x_0)$. Hence, there exists a sole global minimum point x^*. Moreover, x^* is the unique solution of $\nabla F(x) = 0$. Therefore, any accumulation point x^* is the global minimum point, i.e., the sequence $\{x^k\}$ converges to the global minimum point x^*. □

4. Numerical experiments

In this section, some numerical examples are discussed to illustrate the effectiveness and advantages of the proposed two variants of Newton method (denoted as "VNM1" and "VNM2", respectively) for solving the systems of nonlinear equations. We compare the convergence performances of the two variants of Newton method against the Newton method (denoted as "NM") and Halley method (denoted as "Halley") by the iteration step (denoted as "IT"), elapsed CPU time in seconds (denoted as "CPU"), and objective function value (denoted as "Val"). In actual computations, the running is terminated when the current iteration satisfies

$$\text{Val} := \|F(x^k)\| < 10^{-6}$$

or if the number of iteration exceeds the prescribed iteration steps $k_{max} = 100$, where $F(x^k) = \Sigma_{i=1}^{n} f_i^2(x^k)$, $x = (x_1, x_2, \cdots, x_n)^T$, x^k denotes the k-th step iteration in Algorithm 2.3 or Algorithm 2.4.

The numerical experiments have been carried out by MATLAB R2011b 7.1.3 on a PC equipped with an Intel(R) Core(TM) i7-2670QM, CPU running at 2.20 GHZ with 8 GB of RAM in Windows 7 operating system.

Now we perform the following nine test examples. All numerical results are shown in Tables 1-4. In Tables 1-3, we give the different initial guess and the parameters. Obviously, Algorithm 2.3 reduces to the classical Newton method, when we choose the parameters $\alpha = 1,\ \beta = 0,\ \gamma = -1$. Also, Algorithm 2.3 reduces to the Halley method if we take the parameters $\alpha = 2,\ \beta = 2,\ \gamma = 1$.

In Table 4, we demonstrate the numerical performances for six examples from Examples 4.4-4.9. For Example 4.4, we select the initial point with $(0.5, -2)^T$, the parameters $\alpha = 3,\ \beta = 3,\ \gamma = 1.8$ for VNM1 and the parameters $\beta = 3,\ \gamma = 0.9$ for VNM2. For Example 4.5, we set the initial point with $(-1, 10)^T$, the parameters $\alpha = 3,\ \beta = 3,\ \gamma = 3$ for VNM1 and the parameters $\beta = 3,\ \gamma = 3$ for VNM2. For Example 4.6, we choose the initial point with $(2, 3, 3)^T$, the parameters $\alpha = 3,\ \beta = 3,\ \gamma = 2.9$ for VNM1 and the parameters $\beta = 3,\ \gamma = 2.9$ for VNM2. For Example 4.7, we take the initial point with $10^4 \cdot (-10, -2, -3, -0.2)^T$, the parameters $\alpha = 3,\ \beta = 3,\ \gamma = 2$ for VNM1 and the parameters $\beta = 3,\ \gamma = 1.9$ for VNM2. For Example 4.8, we let the initial point with $(10, 10, 10)^T$, the parameters $\alpha = 3,\ \beta = 3,\ \gamma = 2.9$ for VNM1 and the parameters $\beta = 3,\ \gamma = 2.8$ for VNM2. For Example 4.9, we select the initial point with $5 \cdot \text{ones}(n, 1)$, the parameters $\alpha = 3,\ \beta = 3,\ \gamma = 2.7$ for VNM1 and the parameters $\beta = 3,\ \gamma = 2.9$ for VNM2.

From these tables, we can see that both iterative numbers and elapsed CPU times of VNM1 and VNM2 are less than those of the Newton method and Halley method in many case. The reason why the two variants of Newton method are much more efficient than Newton method and Halley method may be the more flexible and widespread selection for parameters.

Example 4.1 ([15]). We consider the system of nonlinear equations (1.1) with the following form:

$$f(x) = \begin{cases} x_1^3 - 3x_1x_2^2 - 1 = 0, \\ 3x_2x_1^2 - x_2^3 + 1 = 0. \end{cases}$$

The exact solution is $x^* = (-0.290514555507, 1.0842150814913)^T$.

Example 4.2 ([5]). We consider the system of nonlinear equations (1.1) with the following form:

$$f(x) = \begin{cases} 3x_1 - \cos(x_2x_3) - 5 = 0, \\ x_1^3 - 81(x_2 + 0.1)^2 + \sin x_3 + 1.06 = 0, \\ e^{-x_2x_3} + 20x_3 + \frac{10\pi - 3}{3} = 0. \end{cases}$$

The exact solution is $x^* = (1.998779542323100, 0.161973550312679, -0.528065506910533)^T$.

Example 4.3 ([19]). We consider the system of nonlinear equations (1.1) with the following form:

$$f(x) = \begin{cases} (x_1 - 5x_2)^2 + 40\sin^2(10x_3) = 0, \\ (x_2 - 2x_3)^2 + 40\sin^2(10x_1) = 0, \\ (3x_1 + x_2)^2 + 40\sin^2(10x_2) = 0. \end{cases}$$

The exact solution is $x^* = (0, 0, 0)^T$.

Example 4.4 ([22]). We consider the system of nonlinear equations (1.1) with the following form:

$$f(x) = \begin{cases} -13 + x_1 + ((5 - x_2)x_2 - 2)x_2 = 0, \\ -29 + x_1 + ((x_2 + 1)x_2 - 14)x_2 = 0. \end{cases}$$

The exact solution is $x^* = (5, 4)^T$.

Example 4.5 ([22]). We consider the system of nonlinear equations (1.1) with the following form:

$$f(x) = \begin{cases} x_1 - e^{x_2} + 1 = 0, \\ x_1 - \cos x_2 - 2 = 0. \end{cases}$$

The exact solution is $x^* = (1.341176629595537, 0.850614425996447)^T$.

Example 4.6 ([6]). We consider the system of nonlinear equations (1.1) with the following form:

$$f(x) = \begin{cases} (x_1 - 1)^4 e^{x_2} = 0, \\ (x_2 - 2)^2(x_1x_2 - 1) = 0, \\ (x_3 + 4)^6 = 0. \end{cases}$$

The exact solution is $x^* = (1, 2, -4)^T$.

Example 4.7 ([12]). We consider the system of nonlinear equations (1.1) with the following form:

$$f(x) = \begin{cases} 10(x_2 - x_1^2) = 0, \\ 1 - x_1 = 0, \\ 90^{\frac{1}{2}}(x_4 - x_3^2) = 0, \\ 1 - x_3 = 0, \\ 10^{\frac{1}{2}}(x_4 + x_2 - 2) = 0, \\ 10^{-\frac{1}{2}}(x_2 - x_4) = 0. \end{cases}$$

The exact solution is $x^* = (1, 1, 1, 1)^T$.

Example 4.8 ([22]). We consider the system of nonlinear equations (1.1) with the following form:

$$f(x) = \begin{cases} x_1^2 + x_2^2 + x_2^3 - x_3 - x_3^2 = 0, \\ 2x_1 + x_2^2 - x_3 = 0, \\ 1 + x_1 - x_2x_3 = 0. \end{cases}$$

The exact solution is $x^* = (-0.717138270295964, -0.203233278645136, -1.393059942219910)^T$.

Example 4.9 ([22]). We consider the system of nonlinear equations (1.1) with the following form:

$$f(x) = \begin{cases} x_i \sin(x_{i+1}) - 1 = 0, \\ x_n \sin(x_1) - 1 = 0, \end{cases}$$

where $n = 16$, $i = 1, 2, \ldots, 15$. The exact solution is $x^* = -1.114157 \cdot (1, 1, \ldots, 1)^T \in \mathbb{R}^{16}$.

Table 1: Numerical results for example 4.1.

Methods		VNM1	VNM2	NM	Halley
Parameters		$\alpha=2,\beta=2,\gamma=2$	$\beta=2,\gamma=2$		
Initials	It	6	6	7	10
$(2,-0.5)^T$	CPU	0.013431	0.011241	0.013870	0.013895
	Val	2.516e − 010	2.516e − 010	1.0195e − 009	1.7879e − 007
	It	34	20	32	26
$(500,50)^T$	CPU	0.017173	0.012883	0.015482	0.014889
	Val	3.4362e − 011	5.0377e − 011	3.3054e − 010	5.1109e − 007
	It	26	26	–	24
$(100,100)^T$	CPU	0.01655	0.014466	–	0.013880
	Val	1.9425e − 010	1.9425e − 010	–	5.3603e − 007

Table 2: Numerical results for Example 4.2.

Methods		VNM1	VNM2	NM	Halley
Parameters		$\alpha=2,\beta=2,\gamma=1.8$	$\beta=2,\gamma=1.8$		
Initials	It	15	11	–	14
$(2,1,1)^T$	CPU	0.019879	0.020301	–	0.023726
	Val	1.2505e − 007	2.9591e − 007	–	3.4397e − 007
	It	15	14	–	22
$(1,0,1)^T$	CPU	0.019571	0.019100	–	0.019471
	Val	9.4170e − 007	5.3605e − 007	–	6.9823e − 007
	It	11	11	–	18
$(10,10,10)^T$	CPU	0.020074	0.018517	–	0.021271
	Val	9.4538e − 007	5.5233e − 007	–	1.6925e − 007

Table 3: Numerical results for Example 4.3.

Methods		VNM1	VNM2	NM	Halley
Parameters		$\alpha=3,\beta=3,\gamma=3$	$\beta=3,\gamma=3$		
Initials	It	4	4	–	12
$(0.1,0.1,0.1)^T$	CPU	0.020039	0.019090	–	0.020479
	Val	1.3221e − 009	1.2899e − 009	–	6.2214e − 007
	It	2	2	10	8
$(0.01,0.01,0.01)^T$	CPU	0.018182	0.027469	0.020194	0.020317
	Val	9.3543e − 010	9.3543e − 010	2.3049e − 007	2.9993e − 007

Table 4: Numerical results for Examples 4.4-4.9.

Examples		VNM1	VNM2	NM	Halley
	It	23	21	–	59
Example 4.4	CPU	0.012871	0.024354	–	0.026726
	Val	3.9611e − 007	2.6462e − 007	–	5.1715e − 007
	It	10	14	26	22
Example 4.5	CPU	0.011445	0.010914	0.011730	0.011648
	Val	1.1887e − 007	6.0749e − 007	2.0876	6.7433e − 007
	It	33	11	100	100
Example 4.6	CPU	0.014556	0.023012	0.018521	0.018835
	Val	2.7697e − 007	2.7240e − 007	2.7136 − 007	1.522e − 007
	It	46	45	60	58
Example 4.7	CPU	0.01534	0.02774	0.01554	0.015613
	Val	3.9978e − 007	2.6487e − 007	1.8625 − 007	5.2191e − 007
	It	12	9	–	23
Example 4.8	CPU	0.01500	0.01189	–	0.015119
	Val	1.9616e − 007	7.2264e − 007	–	4.2569e − 007
	It	10	9	–	–
Example 4.9	CPU	0.021609	0.027686	–	–
	Val	7.7727e − 007	6.6277e − 007	–	–

5. Conclusion

In this paper, two variants of Newton iteration method are investigated for solving the systems of nonlinear equations. These approaches can be regarded as the generalized forms of the classical Newton method and the Halley method. The proposed approaches are illustrated by some numerical examples and compared with the classical Newton method and the Halley method. Numerical test results demonstrate that two variants of Newton iteration method are very efficient.

Acknowledgment

The authors thank the anonymous referee for helping to improve the original manuscript by valuable suggestions. The research was supported by Fujian Natural Science Foundation (2015J01578,2016J01005), The Outstanding Talents in The New Century Training Plan of Fujian Province University (15kxtzl3,2015; 17kxtzl9,2017), National Postdoctoral Program for Innovative Talents (BX201600182) and China Postdoctoral Science Foundation (2016M600141), Strategic Priority Research Program (B) of the Chinese Academy of Sciences (XDB18010202).

References

[1] H.-B. An, Z.-Z. Bai, *A globally convergent Newton-GMRES method for large sparse systems of nonlinear equations*, Appl. Numer. Math., **57** (2007), 235–252. 1

[2] Z.-Z. Bai, *A class of two-stage iterative methods for systems of weakly nonlinear equations*, Numer. Algorithms, **14** (1997), 295–319. 1

[3] Z.-Z. Bai, V. Migallón, J. Penadés, D. B. Szyld, *Block and asynchronous two-stage methods for mildly nonlinear systems*, Numer. Math., **82** (1999), 1–20. 1

[4] Z.-Z. Bai, X. Yang, *On HSS-based iteration methods for weakly nonlinear systems*, Appl. Numer. Math., **59** (2009), 2923–2936. 1

[5] R. L. Burden, J. D. Faires, *Numerical analysis*, Thomson Learning, New York, (2001). 4.2

[6] J. E. Dennis, R. B. Schnabel, *Numerical methods for unconstrained optimization and nonlinear equation*, Prentice Hall, New Jersey, (1983). 4.6

[7] F. Gao, X.-J. Yang, Y.-F. Zhang, *Exact traveling wave solutions for a new nonlinear heat-transfer equation*, Thermal Science, **21** (2016), 309–315. 1

[8] N. Huang, C.-F. Ma, *A regularized smoothing Newton method for solving SOCCPs based on a new smoothing C-function*, Appl. Math. Comput., **230** (2014), 315–329. 1

[9] C. T. Kelley, *Iterative methods for linear and nonlinear equations*, Society for Industrial and Applied Mathematics (SIAM), Philadelphia, (1995). 3

[10] C. T. Kelley, *Solving Nonlinear Equations with Newton's Method*, Society for Industrial and Applied Mathematics (SIAM), Philadelphia, (2003). 1

[11] Y. Levin, A. Ben-Israelb, *Directional Halley and Quasi-Halley Methods in n Variables*, Stud. Comput. Math., Amsterdam, (2001). 1, 1

[12] J. J. Moré, B. S. Grabow, K. E. Hillstrom, *Testing unconstrained optimization software*, ACM Trans. Math. Software, **7** (1981), 17–41. 4.7

[13] M. Narang, S. Bhatia, V. Kanwar, *New two-parameter Chebyshev-Halley-like family of fourth and sixthorder methods for systems of nonlinear equations*, Appl. Math. Comput., **275** (2016), 394–403. 1

[14] Y. Narushima, N. Sagara, H. Ogasawara, *A smoothing Newton method with Fischer-Burmeister function for second-order cone complementarity problems*, J. Optim. Theory Appl., **149** (2011), 79–101. 1

[15] G. H. Nedzhibov, *A family of multi-pointiterative methods for solving systems of nonlinear equations*, J. Comput. Appl. Math., **222** (2008), 244–250. 1, 4.1

[16] M. A. Noor, *New classes of iterative methods for nonlinear equations*, Appl. Math. Comput., **191** (2007), 128–131. 1

[17] M. A. Noor, F. A. Shah, *A Family of Iterative Schemes for Finding Zeros of Nonlinear Equations having Unknown Multiplicity*, Appl. Math. Inf. Sci., **8** (2014), 2367–2373. 1

[18] F. A. Shah, M. A. Noor, *Higher order iterative schemes for nonlinear equations using decomposition technique*, Appl. Math. Comput., **266** (2015), 414–423. 1

[19] J. Z. Sun, J. K. Zhang, *Solving nonlinear systems of equations based on social cognitive optimization*, Comput. Eng. Appl., **44** (2008), 42–46. 4.3

[20] J. F. Traub, *Iterative Methods for the Solution of Equations*, Prentice Hall, Englewood, (1964). 1

[21] M. A. Woodbury, *Inverting modified matrices*, Princeton University, Princeton, (1950). 3

[22] X. Xiao, H. Yin, *A new class of methods with higher order of convergence for solving systems of nonlinear equations*, Appl. Math. Comput., **264** (2015), 300–309. 1, 4.4, 4.5, 4.8, 4.9

[23] X.-J. Yang, *A new integral transform with an application in heat-transfer problem*, Thermal Sci., **20** (2016), 677–681. 1

[24] X.-J. Yang, F. Gao, *New Technology for Solving Diffusion and Heat Equations*, Thermal Sci., **21** (2017), 133–140. 1