Commun. Math. Sci.
Vol. 21, No. 2, pp. 543–557


# A DERIVATIVE-FREE CONJUGATE GRADIENT METHOD FOR LARGE-SCALE NONLINEAR SYSTEMS OF MONOTONE EQUATIONS*

JING GAO†, YANRAN LI‡, MINGYUAN CAO§, YUETING YANG¶, AND XUE BAI‖

**Abstract.** This paper presents a derivative-free conjugate gradient type algorithm for large-scale nonlinear systems of monotone equations. New search directions with superior numerical performance are constructed by introducing a new conjugate parameter and particular spectral parameters. These search directions inherit the numerical stability of RMIL search direction and satisfy the sufficient descent condition independent of step size. The method combines the hyperplane projection and the derivative-free line search technique to compute the iteration points. Under some appropriate assumptions, the global convergence of the given methods is established. Numerical experiments indicate that the proposed algorithms are effective.

**Keywords.** Derivative-free technique; nonlinear systems of monotone equations; projection technology; conjugate gradient method.

**AMS subject classifications.** 90C06; 90C56; 65K05; 65K10.

## 1. Introduction

Nonlinear systems of monotone equations widely exist in image segmentation [1], operation and control of power system [2], signal reconstruction [3] and many other fields. Many problems of relevance in monotone variational inequality problems [4] and the subproblems in the generalized proximal algorithms with Bregman distances [5] can be transformed into the following nonlinear equations

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

where $F(x):\mathbb{R}^n\to\mathbb{R}^n$ is monotone and continuous, which means $F(x)$ satisfies the inequality $(F(x)-F(y))^{\mathrm{T}}(x-y)\geq 0$ for all $\forall x,y\in\mathbb{R}^n$.

Among the iterative methods for solving (1.1), the Newton method, quasi-Newton method [6], and their variants are very popular because of their fast local superlinear convergence property. However, they are not suitable for solving large-scale nonlinear equations because they need to solve linear equations using the Jacobian matrix or an approximation of it in each iteration. Therefore, many scholars prefer to use the conjugate gradient method with a simple structure and low memory requirement to solve large-scale nonlinear monotone equations [7–12].

Rivaie et al. [13] proposed RMIL conjugate gradient method to solve the unconstrained optimization problem. The numerical results show that RMIL conjugate gradient method has better performance than other conjugate gradient methods. La Cruz and

*Received: September 05, 2021; Accepted (in revised form): June 20, 2022. Communicated by Wotao Yin.
†School of Mathematics and Statistics, Beihua University, Jilin, 132013, P.R. China (gaojingjy@163.com).
‡School of Mathematics and Statistics, Beihua University, Jilin, 132013, P.R. China (2271152741@qq.com).
§Corresponding author. School of Mathematics and Statistics, Beihua University, Jilin, 132013, P.R. China (caomingyuan0918@163.com).
¶School of Mathematics and Statistics, Beihua University, Jilin, 132013, P.R. China; School of Science, Jiangsu University, Zhenjiang, Jiangsu, 212013, P.R. China (yyt2858@163.com).
‖School of Mathematics and Statistics, Beihua University, Jilin, 132013, P.R. China (1967288490@qq.com).

Raydan [14] presented the spectral algorithm for nonlinear equations. Li [15] presents a class of derivative-free methods for solving nonlinear monotone equations based on the conjugate gradient method and line search technique. Ahookhosh [16] proposed two derivative-free conjugate gradient methods for solving nonlinear monotone equations. Recently, Fang [17] proposed an improved RMIL derivative-free conjugate gradient method for solving nonlinear monotone equations based on RMIL conjugate gradient method.

Liu and Feng [18] presented a class of derivative-free conjugate gradient methods for solving nonlinear monotone equations with convex constraints and improved DY conjugate parameters. In this way, the algorithm inherited the stability of DY conjugate gradient method while ensuring the sufficient descent of the search direction and the global convergence of the algorithm.

In order to make the algorithm quickly converge to the solution $x^*$ of the nonlinear monotone system (1.1), according to the monotonicity of $F(x)$, the selection of iteration points can be intervened by using the projection technique and appropriate linear search strategy. For example, Solodov and Svaiter [19] combined an inexact Newton method with a projection technique for solving nonlinear monotone equations. Zhang [20] combined the spectral gradient method with the projection method to solve nonlinear monotone equations.

Inspired by the above literature, this paper propose a derivative-free conjugate gradient method for solving nonlinear monotone equations, improves the conjugate parameter, and uses the inequality proof technique to construct the corresponding spectral parameter, so that the search direction satisfies the sufficient descent condition and bounded property, and then proves the global convergence of the algorithm. The main innovations of the algorithm include:

- Improved conjugate parameter and spectral parameters, three new derivative-free conjugate gradient directions are proposed, which inherit the theoretical properties of RMIL conjugate gradient directions and satisfy the sufficient descent condition and boundedness.
- A derivative-free conjugate gradient algorithm with global convergence is designed, which inherits the numerical stability and superiority of the RMIL conjugate gradient method for solving large-scale nonlinear equations, and is superior to the improved RMIL conjugate gradient method proposed by [17].

The rest of this paper is organized as follows. In Section 2, we will introduce the basic principle, concrete steps, and sufficient descent proof of the proposed new derivative-free conjugate gradient algorithm. In Section 3, we will give the boundedness of search direction and the global convergence proof of the algorithm. In Section 4, we will analyze the numerical performance of the algorithm, including the numerical results of solving large-scale nonlinear monotone equations and the comparison with other algorithms.

## 2. Algorithm

In this section, our main aim is to propose a new derivative-free conjugate gradient method for solving monotone Equation (1.1). We define a new conjugate parameter

$$\beta_k^N = \frac{F_k^{\mathrm{T}} y_{k-1}}{d_{k-1}^{\mathrm{T}} w_{k-1}}, \tag{2.1}$$

in which $F_k$ denotes the value of $F(x)$ at current point $x_k$, $d_{k-1}$ denotes the search direction at $x_{k-1}$, $y_{k-1} = F_k - F_{k-1}$, $w_{k-1} = d_{k-1} + t_{k-1} y_{k-1}$ and there exists $t \in (0,1)$

satisfying

$$t_{k-1}=\begin{cases} t, & \text{if } d_{k-1}^{\mathrm{T}}y_{k-1}\geq 0,\\ -t, & \text{if } d_{k-1}^{\mathrm{T}}y_{k-1}<0.\end{cases}$$

In order to obtain the search direction with sufficient descent property, inspired by [17], we propose three spectral parameters based on the conjugate parameter (2.1). On the one hand, we introduce the spectral parameters $\theta_k^1$ and $\theta_k^2$, combining $\beta_k^N$ to construct search directions $d_k^1$ and $d_k^2$ as follows

$$d_k^{1,2}=\begin{cases} -F_k, & \text{if } k=0,\\ -\theta_k F_k+\beta_k^N d_{k-1}, & \text{if } k\geq 1.\end{cases} \tag{2.2}$$

Here, we choose two different spectral parameters

$$\theta_k^1=1+\frac{(F_k^{\mathrm{T}}y_{k-1})^2\|d_{k-1}\|^2}{4\gamma(d_{k-1}^{\mathrm{T}}w_{k-1})^2\|F_k\|^2}, \tag{2.3}$$

$$\theta_k^2=1+\frac{(F_k^{\mathrm{T}}d_{k-1})^2\|y_{k-1}\|^2}{4\gamma(d_{k-1}^{\mathrm{T}}w_{k-1})^2\|F_k\|^2}. \tag{2.4}$$

On the other hand, we combine parameter

$$\theta_k^3=\frac{(F_k^{\mathrm{T}}y_{k-1})^2\|d_{k-1}\|^2}{4\gamma(d_{k-1}^{\mathrm{T}}w_{k-1})^2}, \tag{2.5}$$

with $\beta_k^N$ to construct search direction

$$d_k^3=\begin{cases} -F_k, & \text{if } k=0,\\ -F_k+\beta_k^N d_{k-1}-\theta_k^3 y_{k-1}, & \text{if } k\geq 1.\end{cases} \tag{2.6}$$

In the formulae (2.3), (2.4) and (2.5), we set parameter $\gamma\in(0,1)$. Search directions $d_k^1$, $d_k^2$ and $d_k^3$ have similar theoretical properties, such as descent and boundedness, but they are not consistent in numerical performance. Numerical comparisons are listed in Section 4.

Now, we will select an appropriate line search technique to get the steplength $\alpha_k$. Zhang and Zhou [20] presented the following derivative-free line search rule, which requires $\alpha_k$ to be the largest steplength of $\alpha_k=\max\{s,\rho s,\rho^2 s,\ldots\}$ to satisfy the following condition

$$-F(x_k+\alpha_k d_k)^{\mathrm{T}}d_k\geq\sigma\alpha_k||F(x_k+\alpha_k d_k)||||d_k||^2, \tag{2.7}$$

where $\sigma>0, s>0$ and $0<\rho<1$.

After the search direction $d_k$ and steplength $\alpha_k$ are obtained, we set auxiliary point

$$z_k=x_k+\alpha_k d_k, \tag{2.8}$$

which satisfies

$$F(z_k)^{\mathrm{T}}(x_k-z_k)>0. \tag{2.9}$$

For any $x^*$ such that $F(x^*)=0$, by the monotonicity of $F(x)$, we have

$$F(z_k)^{\mathrm{T}}(x^*-z_k)=-(F(x^*)-F(z_k))^{\mathrm{T}}(x^*-z_k)\leq 0.$$

Thus the hyperplane

$$H_k=\{x\in\mathbb{R}^n|F(z_k)^{\mathrm{T}}(x-z_k)=0\}$$

strictly separates the current iterate $x_k$ from zeros of the equation system (1.1).

Based on the above conclusion, Solodov and Svaiter [19] projected $x_k$ onto $H_k$ to get the next iteration point $x_{k+1}$. Its iteration format is

$$x_{k+1}=x_k-\frac{F(z_k)^{\mathrm{T}}(x_k-z_k)}{||F(z_k)||^2}F(z_k). \tag{2.10}$$

This projection technique combined with the derivative-free line search technique ensures the boundedness of the search direction and the global convergence of the algorithm. Numerical experiments show that this method is superior to other similar algorithms in solving large-scale nonlinear monotone equations. Next, we give the specific steps of the new algorithm.

ALGORITHM 2.1.

*Step 0: Given $x_0\in\mathbb{R}^n$, choose $\delta>0$, $\sigma>0$, $s>0$, $\epsilon>0$, $0<\rho<1$, $k_{\max}>0$, $d_0=-F_0$. Set $k:=0$.*

*Step 1: If $||F_k||<\epsilon$ and $k<k_{\max}$, the algorithm is terminated, output $x_k$; otherwise set $\alpha_k=s$ and go to Step 2.*

*Step 2: If $\alpha_k$ satisfy (2.7), then set auxiliary point $z_k=x_k+\alpha_k d_k$, go to Step 3, otherwise, set $\alpha_k\leftarrow\rho\alpha_k$, go to Step 2.*

*Step 3: Compute $F(z_k)$, if $\|F(z_k)\|\leq\epsilon$, algorithm stops; otherwise, use (2.10) to calculate $x_{k+1}$ go to Step 4.*

*Step 4: Compute $F(x_{k+1})$, $y_k=F(x_{k+1})-F(x_k)$, $w_k=d_k+t_k y_k$. According to (2.1), (2.2) and (2.6), determine the search direction $d_{k+1}$. Set $k:=k+1$ and go to Step 1.*

Here, we will prove that three directions $d_k^1$, $d_k^2$ and $d_k^3$ satisfy the sufficient descent condition.

LEMMA 2.1. *Let the sequence $\{d_k\}$ be generated by Algorithm 2.1, then $d_k$ satisfies the sufficient descent condition*

$$F_k^{\mathrm{T}}d_k\leq-\delta\|F_k\|^2,\ \delta>0. \tag{2.11}$$

*Proof.*

(1) When $\theta$ is defined by $\theta_k^1$ , using (2.1), (2.2) and (2.3), we set $u=\sqrt{2\gamma}d_{k-1}^{\mathrm{T}}w_{k-1}F_k$, $v=\frac{1}{\sqrt{2\gamma}}F_k^{\mathrm{T}}y_{k-1}d_{k-1}$ and use $u^{\mathrm{T}}v\leq\frac{1}{2}(||u||^2+||v||^2)$, then, for $k\in\mathbb{N}$, we have

$$\begin{aligned}F_k^{\mathrm{T}}d_k &= F_k^{\mathrm{T}}\left(-\theta_k F_k+\frac{F_k^{\mathrm{T}}y_{k-1}}{d_{k-1}^{\mathrm{T}}w_{k-1}}d_{k-1}\right)\\ &= -\left[1+\frac{(F_k^{\mathrm{T}}y_{k-1})^2||d_{k-1}||^2}{4\gamma||F_k||^2(d_{k-1}^{\mathrm{T}}w_{k-1})^2}\right]||F_k||^2+\frac{F_k^{\mathrm{T}}y_{k-1}F_k^{\mathrm{T}}d_{k-1}}{d_{k-1}^{\mathrm{T}}w_{k-1}}\end{aligned}$$

$$
\begin{aligned}
&= -\|F_k\|^2 + \frac{F_k^{\mathrm{T}} y_{k-1} F_k^{\mathrm{T}} d_{k-1} \|F_k\|^2 d_{k-1}^{\mathrm{T}} w_{k-1} - \frac{1}{4\gamma}(F_k^{\mathrm{T}} y_{k-1})^2 \|d_{k-1}\|^2 \|F_k\|^2}{\|F_k\|^2 (d_{k-1}^{\mathrm{T}} w_{k-1})^2} \\
&\leq -\|F_k\|^2 + \frac{\frac{1}{2}(2\gamma)(d_{k-1}^{\mathrm{T}} w_{k-1})^2 ||F_k||^2}{(d_{k-1}^{\mathrm{T}} w_{k-1})^2} \\
&= -(1-\gamma)||F_k||^2. \qquad (2.12)
\end{aligned}
$$

(2) When $\theta$ is defined by $\theta_k^2$, using (2.1), (2.2) and (2.4), we set $u=\sqrt{2\gamma} d_{k-1}^{\mathrm{T}} w_{k-1} F_k$, $v=\frac{1}{\sqrt{2\gamma}} F_k^{\mathrm{T}} d_{k-1} y_{k-1}$, then, for $k \in \mathbb{N}$ , we have

$$
\begin{aligned}
F_k^{\mathrm{T}} d_k &= F_k^{\mathrm{T}} \left( -\theta_k F_k + \frac{F_k^{\mathrm{T}} y_{k-1}}{d_{k-1}^{\mathrm{T}} w_{k-1}} d_{k-1} \right) \\
&= -\left[ 1 + \frac{(F_k^{\mathrm{T}} d_{k-1})^2 ||y_{k-1}||^2}{4\gamma ||F_k||^2 (d_{k-1}^{\mathrm{T}} w_{k-1})^2} \right] ||F_k||^2 + \frac{F_k^{\mathrm{T}} y_{k-1} F_k^{\mathrm{T}} d_{k-1}}{d_{k-1}^{\mathrm{T}} w_{k-1}} \\
&= -\|F_k\|^2 + \frac{F_k^{\mathrm{T}} y_{k-1} F_k^{\mathrm{T}} d_{k-1} d_{k-1}^{\mathrm{T}} w_{k-1} - \frac{1}{4\gamma}(F_k^{\mathrm{T}} d_{k-1})^2 \|y_{k-1}\|^2}{(d_{k-1}^{\mathrm{T}} w_{k-1})^2} \\
&\leq -\|F_k\|^2 + \frac{\frac{1}{2}(2\gamma)(d_{k-1}^{\mathrm{T}} w_{k-1})^2 ||F_k||^2}{(d_{k-1}^{\mathrm{T}} w_{k-1})^2} \\
&= -(1-\gamma)||F_k||^2. \qquad (2.13)
\end{aligned}
$$

(3) When $d_k$ is determined by (2.5) and (2.6), set $u=\sqrt{2\gamma} d_{k-1}^{\mathrm{T}} w_{k-1} F_k$, $v= \frac{1}{\sqrt{2\gamma}} F_k^{\mathrm{T}} y_{k-1} d_{k-1}$, then, for $k \in \mathbb{N}$, we have

$$
\begin{aligned}
F_k^{\mathrm{T}} d_k &= F_k^{\mathrm{T}} \left[ -F_k + \frac{F_k^{\mathrm{T}} y_{k-1}}{d_{k-1}^{\mathrm{T}} w_{k-1}} d_{k-1} - \frac{(F_k^{\mathrm{T}} y_{k-1})^2 \|d_{k-1}\|^2}{4\gamma (d_{k-1}^{\mathrm{T}} w_{k-1})^2} y_{k-1} \right] \\
&= -\|F_k\|^2 + \frac{F_k^{\mathrm{T}} y_{k-1} F_k^{\mathrm{T}} d_{k-1} d_{k-1}^{\mathrm{T}} w_{k-1} - \frac{1}{4\gamma}(F_k^{\mathrm{T}} y_{k-1})^2 \|d_{k-1}\|^2}{(d_{k-1}^{\mathrm{T}} w_{k-1})^2} \\
&\leq -(1-\gamma)||F_k||^2.
\end{aligned}
$$

Set $\delta = 1-\gamma$, since $\gamma \in (0,1)$, for the above three directions, the sufficient descent conditions are all satisfied.

(4) If $k=0$ , using(2.2), we get $\|F_0^{\mathrm{T}} d_0\| = -\|F_0\|^2$. □

Therefore, the search direction $d_k$ generated by the Algorithm 2.1 satisfies the sufficient descent condition.

## 3. Convergence analysis

In order to prove the global convergence of Algorithm 2.1, we give the following assumptions.

Assumption 3.1.

(1) *The solution set of the system of monotone Equations (1.1) is nonempty.*

(2) *$F(x)$ is Lipschitz continuous on $\mathbb{R}^n$, namely*

$$\|F(y) - F(x)\| \leq L\|y - x\|, \ \forall x, y \in \mathbb{R}^n, \qquad (3.1)$$

*where $L > 0$ is a positive constant.*

Assumption 3.1 implies that

$$\|F(x)\| \le \kappa, \ \forall x \in \mathbb{R}^n, \tag{3.2}$$

where $\kappa$ is a positive constant.

LEMMA 3.1. *Suppose Assumption 3.1 is satisfied and the sequence $\{x_k\}$ is generated by Algorithm 2.1. For any $x^*$ such that $F(x^*) = 0$, we have*

$$\|x_{k+1} - x^*\|^2 + \|x_{k+1} - x_k\|^2 \le \|x_k - x^*\|^2. \tag{3.3}$$

*In addition, the sequence $\{x_k\}$ satisfies*

$$\lim_{k\to\infty} \|x_{k+1} - x_k\| = 0. \tag{3.4}$$

The proof can be referenced from [19].

REMARK 3.1. From (2.10) and (2.7), we have

$$\|x_{k+1} - x_k\| = \frac{\alpha_k |F^{\mathrm{T}}(z_k) d_k| \|F(z_k)\|}{\|F(z_k)\|^2} \ge \sigma \|\alpha_k d_k\|^2. \tag{3.5}$$

Using (3.4), we can get

$$\lim_{k\to\infty} \|\alpha_k d_k\| = 0. \tag{3.6}$$

LEMMA 3.2. *Suppose Assumption 3.1 is satisfied and the sequences $\{x_k\}, \{d_k\}$ are generated by Algorithm 2.1. Then we have*

$$\|d_k\| \le \kappa \left[1 + Ls + \frac{(Ls)^2}{4\gamma}\right], \tag{3.7}$$

*where $\kappa > 0, 0 < \gamma < 1, s > 0, L > 0$.*

*Proof.* From (2.10) and Step 2 of Algorithm 2.1, we have

$$\|x_{k+1} - x_k\| = \frac{\|F(z_k)^{\mathrm{T}}(x_k - z_k) F(z_k)\|}{\|F(z_k)\|^2} \le \|x_k - z_k\| = \alpha_k \|d_k\|. \tag{3.8}$$

According to the definition of line search in Step 2 of Algorithm 2.1 and $\alpha_k = \max\left\{s, \rho s, \rho^2 s, \cdots\right\}$, $0 < \rho < 1$, we get

$$\alpha_k \le s. \tag{3.9}$$

From the definition of $w_{k-1}$ , we obtain $t_{k-1} d_{k-1}^{\mathrm{T}} y_{k-1} \ge 0$ , and then

$$d_{k-1}^{\mathrm{T}} w_{k-1} \ge \|d_{k-1}\|^2. \tag{3.10}$$

Therefore, for $\theta = \theta_k^1$ , from (3.1), (3.2), (3.8), (3.9) and (3.10), we have

$$\begin{aligned} \|d_k\| &= \left\| -\left[1 + \frac{\left(F_k^{\mathrm{T}} y_{k-1}\right)^2 \|d_{k-1}\|^2}{4\gamma \left(d_{k-1}^{\mathrm{T}} w_{k-1}\right)^2 \|F_k\|^2}\right] F_k + \frac{F_k^{\mathrm{T}} y_{k-1}}{d_{k-1}^{\mathrm{T}} w_{k-1}} d_{k-1} \right\| \\ &\le \|F_k\| \left[1 + \frac{||y_{k-1}||^2 ||d_{k-1}||^2}{4\gamma (d_{k-1}^{\mathrm{T}} w_{k-1})^2} + \frac{||y_{k-1}|| ||d_{k-1}||}{d_{k-1}^{\mathrm{T}} w_{k-1}}\right] \end{aligned}$$

$$\leq \|F_k\|\left(1+\frac{\|y_{k-1}\|^2}{4\gamma\|d_{k-1}\|^2}+\frac{\|y_{k-1}\|}{\|d_{k-1}\|}\right)$$
$$\leq \|F_k\|\left(1+\frac{L^2\alpha_{k-1}^2\|d_{k-1}\|^2}{4\gamma\|d_{k-1}\|^2}+\frac{L\alpha_{k-1}\|d_{k-1}\|}{\|d_{k-1}\|}\right)$$
$$\leq \kappa\left[1+Ls+\frac{(Ls)^2}{4\gamma}\right].$$

Similarly, when $\theta=\theta_k^2$ and $d_k=d_k^3$, the conclusions also hold. □

LEMMA 3.3. *Suppose Assumption 3.1 is satisfied and the sequences* $\{x_k\},\{d_k\}$ *are generated by Algorithm 2.1. If there exists a constant* $\epsilon$*, such that* $||F_k||\geq\epsilon$ *for all* $k\in\mathbb{N}\cup\{0\}$*, then we have*

$$\alpha_k\geq\min\left\{s,\frac{\delta\epsilon^2}{\rho^{-1}\kappa^2\left\{L+\sigma\kappa\rho^{-1}\left[\rho+Ls+(Ls)^2+\frac{(Ls)^3}{4\gamma}\right]\right\}\left[1+Ls+\frac{(Ls)^2}{4\gamma}\right]^2}\right\}. \tag{3.11}$$

The proof is analogous to that of Lemma 3.3 in [17]. We omit the proof here.

THEOREM 3.1. *Suppose Assumption 3.1 is satisfied, and the sequences* $\{x_k\},\{d_k\}$ *are generated by Algorithm 2.1, then we have*

$$\liminf_{k\to\infty}\|F_k\|=0. \tag{3.12}$$

*In particular, the sequence* $\{x_k\}$ *converges to* $x^*$ *and* $F(x^*)=0$*.*

*Proof.* If (3.12) does not hold, then there exists a constant $\epsilon>0$ such that

$$\|F_k\|\geq\epsilon,\quad \forall k\geq 0. \tag{3.13}$$

From Lemma 2.1, we get

$$||F_k||||d_k||\geq||F_k^{\mathrm{T}}d_k||\geq(1-\gamma)||F_k||^2. \tag{3.14}$$

From (3.13) and (3.14), we have

$$||d_k||\geq(1-\gamma)||F_k||\geq\delta\epsilon. \tag{3.15}$$

Suppose $\widetilde{x}$ is an arbitrary accumulation point of $\{x_k\}$ and $K_1$ is an infinite index set such that

$$\lim_{k\in K_1,k\to\infty}x_k=\widetilde{x}. \tag{3.16}$$

From (3.4), (3.8) and (3.16), we can get

$$\lim_{k\in K_1,k\to\infty}\alpha_k||d_k||=0. \tag{3.17}$$

On the other hand, combined with Lemma 3.3 and (3.15),

$$\alpha_k\|d_k\|\geq\min\left\{\delta\epsilon s,\frac{\delta^2\epsilon^3}{\rho^{-1}\kappa^2\left\{L+\sigma\kappa\rho^{-1}\left[\rho+Ls+(Ls)^2+\frac{(Ls)^3}{4\gamma}\right]\right\}\left[1+Ls+\frac{(Ls)^2}{4\gamma}\right]^2}\right\}$$

$> 0.$

This inequality and (3.17) are a contradiction, then the conclusion (3.12) holds. From Assumption 3.1, Lemma 3.1 and (3.12), we see that the sequence $\{x_k\}$ converges to some accumulation point $x^*$ such that $F(x^*)=0$. ◻

In order to further prove the R-linear convergence rate of the algorithm, we need to give the following assumptions.

ASSUMPTION 3.2. *For $\forall \tilde{x}\in\Omega$, there exist $\varrho\in(0,1)$ and $\nu>0$ satisfying*

$$\varrho dist(x,\Omega)\le\|F_k\|^2, \forall x\in N(\tilde{x},\nu), \tag{3.18}$$

*where $\Omega$ is the solution set of problem (1.1), $N(\tilde{x},\nu)=\{x\in\mathbb{R}^n \mid \|x-\tilde{x}\|\le\nu\}$, $dist(x,\Omega)$ represents the distance from point $x$ to solution set $\Omega$.*

THEOREM 3.2. *Suppose Assumption 3.1 and Assumption 3.2 are satisfied, the sequence $\{x_k\}$ is generated by Algorithm 2.1, then the sequence $\{dist(x,\Omega)\}$ converges Q-linearly to 0, therefore $\{x_k\}$ is R-linear convergent.*

*Proof.* Set $\omega_k=\operatorname{argmin}\{\|x_k-\omega\| \mid \omega\in\Omega\}$, we know that $\omega_k$ is the closest solution to point $x_k$ in the solution set $\Omega$, that is

$$\mathrm{dist}(x_k,\Omega)=\|x_k-\omega_k\|. \tag{3.19}$$

Using (2.11) and Cauchy-Schwartz inequality, we can get

$$\|d_k\|\ge\delta\|F_k\|. \tag{3.20}$$

Since $\omega_k\in\Omega$, from (3.3), (3.18), (3.19) and (3.20), we get

$$\begin{aligned}
\mathrm{dist}(x_{k+1},\Omega)^2 &\le \|x_{k+1}-\omega_k\|^2\\
&\le \|x_k-\omega_k\|^2-\|x_{k+1}-x_k\|^2\\
&= \mathrm{dist}(x_k,\Omega)^2-\|x_{k+1}-x_k\|^2\\
&\le \mathrm{dist}(x_k,\Omega)^2-\sigma^2\|\alpha_k d_k\|^4\\
&\le \mathrm{dist}(x_k,\Omega)^2-\sigma^2\alpha_k^4\delta^4\|F_k\|^4\\
&\le (1-\sigma^2\varrho^2\alpha_k^4\delta^4)\mathrm{dist}(x_k,\Omega)^2.
\end{aligned}$$

It shows the sequence $\{\mathrm{dist}(x_k,\Omega)\}$ converges Q-linearly to 0. Using $1-\sigma^2\varrho^2\alpha_k^4\delta^4\in(0,1)$, then $\{x_k\}$ is R-linear convergent. ◻

## 4. Numerical experiment

In this section, we will compare Algorithm 2.1 with the DFPB1 method in [16] and the MRMIL1 method in [17]. Our tests are implemented in MatlabR2015b, run on a personal computer with 4 GB RAM and Intel CPU I5-4210. We compare all methods, and give the numerical performance comparison chart. We employ the performance profiles [21], which are defined by the following fraction

$$\rho_\nu(\tau)=\frac{1}{|P|}\left|\left\{p\in P: log_2\left(\frac{t_{p,\nu}}{\min\{t_{p,\nu}:\nu\in V\}}\right)\le\tau\right\}\right|$$

to describe the performance of the algorithms. Here P is the test set, |P| is the number of problems in the test set P, V is the set of optimization solvers, and $t_{p,\nu}$ is the

CPU time (or the number of the function evaluations, or the number of iterations) for $p \in P$ and $\nu \in V$. The numerical experiments include 10 large-scale nonlinear monotone equations. The problems 1-8 are selected from problem 1 to 8 in [17], and the problems 9-10 are selected from problem 4 to 5 in [22] with sizes 500, 1000, 3000, 5000, 10000, all problems are initialized with the following 8 starting points: $x_0^1 = (10, 10, \ldots, 10)^{\mathrm{T}}$, $x_0^2 = (-10, -10, \ldots, -10)^{\mathrm{T}}$, $x_0^3 = (1, 1, \ldots, 1)^{\mathrm{T}}$, $x_0^4 = -(1, 1, \ldots, 1)^{\mathrm{T}}$, $x_0^5 = (0.1, 0.1, \ldots, 0.1)^{\mathrm{T}}$, $x_0^6 = (1, \frac{1}{2}, \ldots, \frac{1}{n})^{\mathrm{T}}$, $x_0^7 = (\frac{1}{n}, \frac{2}{n}, \ldots, 1)^{\mathrm{T}}$, $x_0^8 = (\frac{n-1}{n}, \frac{n-2}{n}, \ldots, 0)^{\mathrm{T}}$. For all methods, the stopping criteria are (1)$||F(x_k)|| \le \epsilon$; or (2) $||F(z_k)|| \le \epsilon$; or (3) the number of iterations exceeds $k_{\max}$, where $\epsilon = 10^{-4}$, $k_{\max} = 10^5$, $\rho = 0.7$, $\sigma = 0.3$, $\gamma = \frac{1}{4}$.

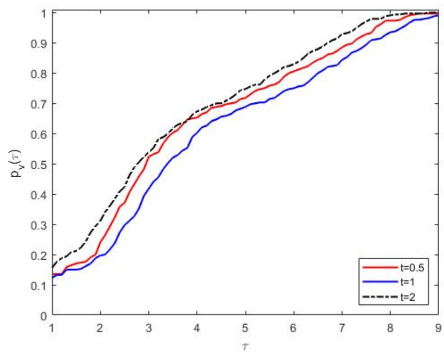


FIG. 4.1. *Performance profiles of Algorithm 2.1 with different t based on the number of function evaluations.*

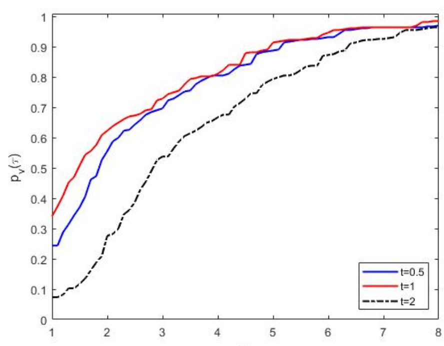


FIG. 4.2. *Performance profiles of Algorithm 2.1 with different t based on the number of iterations.*

We analyze the values of parameters $t, s, \gamma$ in Algorithm 2.1, the value of parameter $t$ in Algorithm 2.1 directly affects the search direction and its descent. Firstly, we study the numerical results of parameter $t = 0.5, 1, 2$ in terms of the number of function evaluations, the number of iterations, and CPU operation time. Figures 4.1-4.3 show the experimental results of the direction with $d_k^1$, $s = 1, \gamma = 0.25$. Obviously, in terms of the number of the function evaluations (see Figure 4.1) and the number of iterations (see Figure 4.2), the three values of $t$ are almost the same, while in terms of CPU operation time (see Figure 4.3), for $t = 1$, it shows a more stable performance.

Secondly, we study the effect of initial steplength $s$ and parameter $\gamma$ on algorithm performance. In Algorithm 2.1, we use $d_k^1$ as the search direction, use the initial step

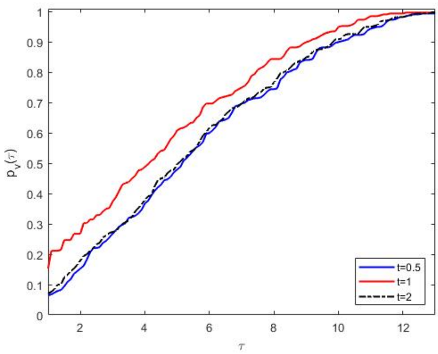


FIG. 4.3. *Performance profiles of Algorithm 2.1 with different t based on the CPU time.*

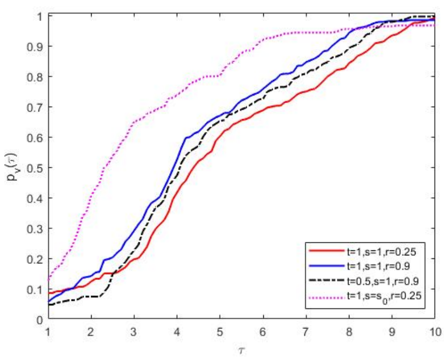


FIG. 4.4. *Performance profiles of Algorithm 2.1 with different s and $\gamma$ based on the number of function evaluations*

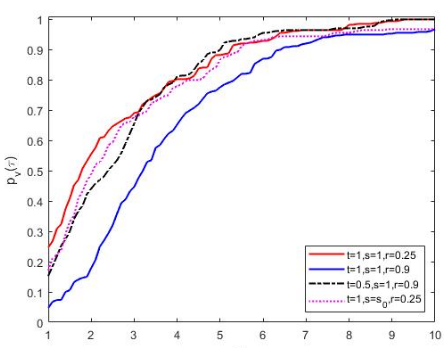


FIG. 4.5. *Performance profiles of Algorithm 2.1 with different s and $\gamma$ based on the number of iterations*

for $s=1$, $s=s_0$, $s_0=\|\frac{F_k^{\mathrm{T}} d_k}{(F(x_k+10^{-8}d_k)-F_k)^{\mathrm{T}} d_k/10^{-8}}\|$, and $\gamma=0.25$ respectively to solve 10 large-scale nonlinear monotone equations, the specific results as shown in Figures 4.4-4.6. Figures 4.1-4.6 show that Algorithm 2.1 with $t=1, s=s_0, \gamma=0,25$ has high stability in function evaluations. However, by observing the numerical results, for solving all test problems, it is found that the average number of function evaluations generated by

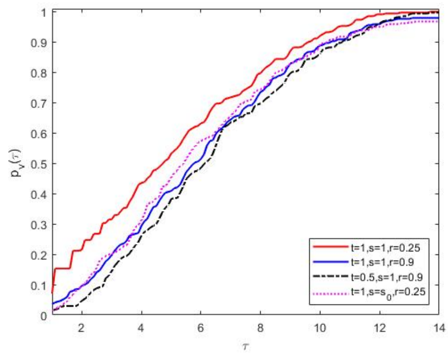


FIG. 4.6. *Performance profiles of Algorithm 2.1 with different $s$ and $\gamma$ based on the CPU time*

Algorithm 2.1 is more than other algorithms, and the CPU operation time is the same as that of other algorithms. Therefore, Algorithm 2.1 is not suitable for selecting initial steplength $s=s_0$. When $t=1, s=1, \gamma=0,25$ (red line), Algorithm 2.1 is almost the same as other algorithms in the number of iterations and function evaluations, but it is in the upstream in terms of the CPU operation time, which shows that in this case, the stability of the algorithm is good, and the average time of solving the test problem is short. Based on the above experimental results, we investigated the numerical performance of three search directions in Algorithm 2.1 under the conditions of $t=1, s=1, \gamma=0.25$, denoted as NA1, NA2, and NA3, as well as DFPB1 and MRMIL1 algorithms proposed by [16] and [17], respectively.

Figures 4.7-4.9 show the performance profiles of the proposed NA1, NA2, and NA3 algorithms compared with MRMIL1 and DFPB1 algorithms in the CPU time, the number of the function evaluations, and the number of iterations. We can see from Figure 4.8 and Figure 4.9 that in terms of the number of iterations and CPU time, the performance curves of Algorithm 2.1 in three directions are significantly higher than that of the other two algorithms. It shows that the new derivative-free conjugate gradient method has good stability in solving large-scale nonlinear monotone equations, can use fewer iterations and CPU time, and has higher efficiency. Specifically, Figure 4.7 shows that the proposed algorithms perform better than DFPB1 and MRMIL1, whereas the NA1 uses slightly fewer function evaluations. In Figure 4.8, the search direction $d_k^3$(NA3) performs better on the number of iterations at $\tau < 2$, and it still has the advantage when $\tau \geq 2$, but it is not evident. Figure 4.9 indicates Algorithm 2.1 proposed in this paper has obvious superiority, and Algorithm 2.1 with $d_k^3$(NA3) is faster in solving large-scale nonlinear monotone equations and has notable advantages among the five algorithms.

In our opinion, the directions of the new algorithms have a natural descending property that does not depend on the step size, and the parameter $t$ is suitably selected so that the function value has sufficient descent along these directions in each iteration. Simultaneously, the formula of the search direction is brief and uses the stored information in the calculation execution process, which results in no significant increase in the amount of calculation and the number of function evaluations. For the above reasons, the numerical performance of the newly proposed three algorithms is superior to that of the other two methods.

Finally, we make a meaningful attempt at line search technology. In [23], Dai

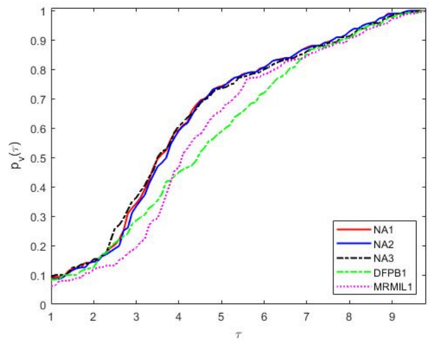


Fig. 4.7. *Performance profiles of different algorithms based on the number of function evaluations*

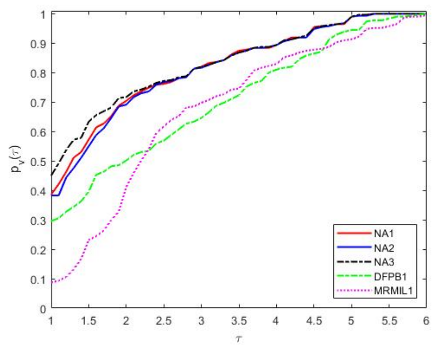


Fig. 4.8. *Performance profiles of different algorithms based on the number of iterations*

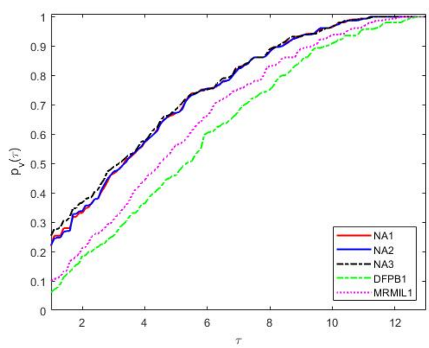


Fig. 4.9. *Performance profiles of different algorithms based on the CPU time*

introduces a line search technique:

$$-F(x_k+\alpha_k d_k)^{\mathrm{T}} d_k \geq \sigma \alpha_k \min\{\|d_k\|^2, \|F(x_k+\alpha_k d_k)\|\|d_k\|^2, -F(x_k)^{\mathrm{T}} d_k\}, \qquad (4.1)$$

$\alpha_k = \max\{s, \rho s, \rho^2 s\}$, $0<\rho<1$, which is used instead of (2.7) to select the appropriate steplength $\alpha_k$. It may find a more accurate $\alpha_k$ to construct an iteration point closer to the real value, which is a technology to reduce the number of iterations and save the operation time; it is also possible that the min function on the right side of the inequality increased the number of function evaluations. The guess is confirmed in the following experiments. We replace the line search technique (2.7) in Algorithm 2.1 with (4.1), in the case of $t=1, s=1, \gamma=0.25, d_k=d_k^1$, then we get Algorithm 4.1.

| | | NA1 | | | MRMIL1 | | | DFPB1 | | | MNA1 | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Prob. | n | nf | iter | CPUt | nf | iter | CPUt | nf | iter | CPUt | nf | iter | CPUt |
| 1 | 500 | 988 | 88 | 0.297 | 992 | 88 | 0.265 | 988 | 86 | 0.295 | 1100 | 97 | 0.316 |
| | 1000 | 1501 | 121 | 0.687 | 1505 | 121 | 0.656 | 1514 | 120 | 0.69 | 1502 | 121 | 0.656 |
| | 3000 | 2904 | 204 | 3.385 | 2908 | 204 | 3.281 | 2914 | 203 | 3.377 | 2934 | 211 | 3.312 |
| | 5000 | 3874 | 259 | 7.548 | 3878 | 259 | 7.295 | 3884 | 258 | 7.613 | 3874 | 259 | 7.389 |
| | 10000 | 5826 | 364 | 22.575 | 5827 | 364 | 22.03 | 5828 | 362 | 22.78 | 5853 | 364 | 24.979 |
| 2 | 500 | 42 | 15 | 0.028 | 46 | 15 | 0.016 | 46 | 14 | 0.021 | 306 | 39 | 0.052 |
| | 1000 | 68 | 18 | 0.035 | 69 | 18 | 0.031 | 70 | 17 | 0.036 | 517 | 58 | 0.111 |
| | 3000 | 143 | 27 | 0.181 | 145 | 27 | 0.172 | 142 | 25 | 0.173 | 950 | 93 | 0.495 |
| | 5000 | 196 | 32 | 0.393 | 197 | 32 | 0.391 | 200 | 31 | 0.413 | 1520 | 135 | 1.32 |
| | 10000 | 304 | 42 | 1.183 | 308 | 42 | 1.171 | 308 | 41 | 1.263 | 2495 | 202 | 4.202 |
| 3 | 500 | 1349 | 715 | 0.703 | 1240 | 685 | 0.641 | 1295 | 711 | 0.669 | **13** | **4** | **0.008** |
| | 1000 | 1319 | 684 | 1.172 | 1316 | 740 | 1.187 | 1365 | 743 | 1.259 | **13** | **4** | **0.01** |
| | 3000 | 1666 | 790 | 4.062 | 1393 | 738 | 3.39 | 1443 | 742 | 3.65 | **13** | **4** | **0.015** |
| | 5000 | 1740 | 797 | 6.975 | 1016 | 474 | 3.952 | 1709 | 676 | 6.523 | **13** | **4** | **0.021** |
| | 10000 | 1582 | 758 | 12.852 | 1590 | 762 | 12.607 | 1613 | 764 | 13.146 | **13** | **4** | **0.035** |
| 4 | 500 | 154 | 28 | 0.063 | 155 | 29 | 0.062 | 154 | 27 | 0.064 | 183 | 34 | 0.097 |
| | 1000 | 236 | 36 | 0.14 | 241 | 37 | 0.125 | 238 | 35 | 0.146 | 248 | 39 | 0.129 |
| | 3000 | 486 | 58 | 0.672 | 492 | 59 | 0.641 | 497 | 58 | 0.701 | 495 | 67 | 0.667 |
| | 5000 | 672 | 73 | 1.526 | 678 | 74 | 1.484 | 684 | 73 | 1.506 | 687 | 77 | 1.598 |
| | 10000 | 1044 | 101 | 4.6 | 1050 | 102 | 4.545 | 1046 | 100 | 4.647 | 1137 | 111 | 4.743 |
| 5 | 500 | 104 | 19 | 0.046 | 115 | 21 | 0.063 | 169 | 30 | 0.267 | 132 | 24 | 0.261 |
| | 1000 | 104 | 19 | 0.156 | 115 | 21 | 0.203 | 169 | 30 | 0.265 | 132 | 24 | 0.269 |
| | 3000 | 104 | 19 | 1.312 | 115 | 21 | 1.579 | 169 | 30 | 2.4 | 132 | 24 | 3.75 |
| | 5000 | 104 | 19 | 3.578 | 115 | 21 | 3.921 | 169 | 30 | 5.753 | 132 | 24 | 6.178 |
| | 10000 | 104 | 19 | 13.726 | 115 | 21 | 16.286 | 169 | 30 | 21.508 | 132 | 24 | 67.4 |
| 6 | 500 | 112 | 17 | 0.047 | 124 | 19 | 0.046 | 82 | 12 | 0.031 | 104 | 17 | 0.047 |
| | 1000 | 133 | 19 | 0.203 | 136 | 20 | 0.171 | 104 | 14 | 0.125 | 156 | 23 | 0.239 |
| | 3000 | 204 | 26 | 2.452 | 205 | 27 | 2.484 | 161 | 18 | 1.906 | 195 | 25 | 2.59 |
| | 5000 | 245 | 29 | 7.717 | 240 | 28 | 7.524 | 201 | 21 | 6.248 | 270 | 36 | 7.757 |
| | 10000 | 329 | 35 | 39.287 | 318 | 33 | 37.944 | 285 | 27 | 34.175 | 358 | 38 | 42.944 |
| 7 | 500 | 46 | 14 | 0.047 | 46 | 14 | 0.016 | 48 | 14 | 0.024 | 280 | 36 | 0.049 |
| | 1000 | 75 | 18 | 0.031 | 70 | 17 | 0.031 | 74 | 17 | 0.041 | 477 | 54 | 0.099 |
| | 3000 | 149 | 26 | 0.187 | 153 | 26 | 0.187 | 159 | 26 | 0.213 | 884 | 87 | 0.484 |
| | 5000 | 214 | 32 | 0.438 | 218 | 32 | 0.422 | 214 | 31 | 0.439 | 1402 | 125 | 1.272 |
| | 10000 | 330 | 43 | 1.281 | 332 | 42 | 1.281 | 335 | 42 | 1.314 | 2326 | 189 | 4.033 |
| 8 | 500 | 121 | 20 | 0.047 | 128 | 22 | 0.046 | 126 | 20 | 0.038 | 114 | 18 | 0.031 |
| | 1000 | 145 | 22 | 0.062 | 155 | 24 | 0.063 | 165 | 24 | 0.071 | 137 | 20 | 0.063 |
| | 3000 | 230 | 29 | 0.234 | 246 | 32 | 0.266 | 657 | 113 | 0.703 | 222 | 32 | 0.234 |
| | 5000 | 283 | 33 | 0.485 | 303 | 37 | 0.499 | 768 | 127 | 1.428 | 275 | 31 | 0.453 |
| | 10000 | 396 | 41 | 1.327 | 410 | 44 | 1.375 | 987 | 156 | 3.477 | 388 | 39 | 1.297 |
| 9 | 500 | 38 | 15 | 0.031 | 34 | 11 | 0.016 | 40 | 15 | 0.018 | 348 | 44 | 0.064 |
| | 1000 | 59 | 18 | 0.032 | 55 | 14 | 0.031 | 66 | 19 | 0.037 | 587 | 65 | 0.132 |
| | 3000 | 129 | 27 | 0.171 | 125 | 23 | 0.172 | 128 | 26 | 0.173 | 1048 | 102 | 0.578 |
| | 5000 | 184 | 33 | 0.391 | 179 | 28 | 0.359 | 187 | 32 | 0.386 | 1676 | 148 | 1.558 |
| | 10000 | 287 | 43 | 1.171 | 282 | 38 | 1.156 | 294 | 43 | 1.189 | 2742 | 221 | 4.894 |
| 10 | 500 | 119 | 21 | 0.046 | 272 | 50 | 0.078 | 1045 | 203 | 0.281 | **19** | **6** | **0.017** |
| | 1000 | 140 | 24 | 0.063 | 319 | 58 | 0.14 | 1441 | 279 | 0.631 | **19** | **6** | **0.007** |
| | 3000 | 178 | 29 | 0.203 | 328 | 57 | 0.344 | 1420 | 272 | 1.655 | **133** | **17** | **0.071** |
| | 5000 | 295 | 48 | 0.547 | 377 | 64 | 0.672 | 1436 | 272 | 2.697 | **146** | **18** | **0.126** |
| | 10000 | 293 | 42 | 1.046 | 345 | 52 | 1.187 | 1549 | 288 | 5.78 | **214** | **23** | **0.345** |

TABLE 4.1. *Numerical Results.*

Algorithm 4.1.

*Step 0: Choose an initial point* $x_0 \in \mathbb{R}^n$ *, choose constants* $\delta > 0, \sigma > 0, s > 0, \epsilon > 0, 1 > \rho > 0, k_{\max} > 0, d_0 = -F(x_0)$. *Set* $k := 0$.

*Step 1: If* $||F_k|| < \epsilon$ *and* $k < k_{\max}$*, the algorithm is terminated, output* $x_k$*; otherwise, set* $\alpha_k = s$.

*Step 2: If* $\alpha_k$ *satisfy (4.1), then set auxiliary point* $z_k = x_k + \alpha_k d_k$*; otherwise, set* $\alpha_k \leftarrow \rho\alpha_k$*, go to Step 2 again.*

*Step 3: Calculate* $F(z_k)$*, if* $\|F(z_k)\| \leq \epsilon$*, then the algorithm stops; otherwise, we use (2.10) to calculate* $x_{k+1}$.

*Step 4: Calculate* $F(x_{k+1})$, $y_k = F(x_{k+1}) - F(x_k), w_k = d_k + t_k y_k$*. We use (2.1), (2.2) and (2.6), to determine the search direction* $d_{k+1}$*. Set* $k := k+1$ *and go to Step 1.*

We used Algorithm 4.1 (MNA1) to solve test problems 1-10, and compared with the numerical results of NA1, MRMIL1, and DFPB1 algorithms, the detailed results are given in Table 4.1. We denoted the number of function evaluations, iterations, and CPU operation time as $nf$, $iter$, and CPUt respectively.

We can see from Table 4.1 that Algorithm 4.1 costs more than twice as much as other algorithms in solving test problems 2, 7, and 9. However, for problems 3 and 10, which are expensive to solve by other algorithms, the optimal solution is obtained in a short time and with a few iterations. It is a surprising result and may be closely related to the structure of the right side of the inequality (4.1), which is worthy of further study and discussion in the following work.

## 5. Conclusion

We mainly study a few efficient algorithms for solving large-scale nonlinear monotone systems in this paper, present a class of derivative-free conjugate gradient methods based on the projection technique, and prove that the search direction satisfies the sufficient descent condition. The global convergence of new algorithms is proved under appropriate assumptions. The numerical results show that the proposed algorithm has high efficiency in solving large-scale nonlinear monotone equations, and its performance is better than other similar algorithms in terms of function evaluations and CPU time.

**Acknowledgments.** This work is supported by the key project of natural science foundation joint fund of Jilin Province (YDZJ202101ZYTS167, YDZJ202201ZYTS303); innovation talent training program of science and technology of Jilin Province (20180519011JH); science and technology development project program of Jilin Province (20190303132SF); youth growth science and technology project of Jilin Province science and technology development plan; the PhD start-up fund of natural science foundation of Beihua University; project of education department of Jilin Province (JJKH202100 30KJ, JJKH20230054KJ).

REFERENCES

[1] Y. Zheng, B. Jeon, D. Xu, Q.M. Wu, and H. Zhang, *Image segmentation by generalized hierarchical fuzzy C-means algorithm*, J. Intell. Fuzzy Syst., 28(2):961–973, 2015. 1

[2] A.J. Wood and B.F. Wollenberg, *Power Generation, Operation, and Control*, John Wiley and Sons, New York, NY, 1984. 1

[3] B. Gu, V.S. Sheng, K.Y. Tay, W. Romano, and S. Li, *Incremental support vector learning for ordinal regression*, IEEE Trans. Neural Netw. Learn. Syst., 26(7):1403–1416, 2015. 1

[4] A.N. Iusem and M.V. Solodov, *Newton-type methods with generalized distance for constrained optimization*, Optimization, 41(3):257–278, 1997. 1

[5] Y.B. Zhao and D. Li, *Monotonicity of fixed point and normal mapping associated with variational inequality and its applications*, SIAM J. Optim., 11(4):962–973, 2011. 1

[6] W.J. Zhou and D.H. Li, *A globally convergent BFGS method for nonlinear monotone equations without any merit functions*, Math. Comput., 77(264):2231–2240, 2008. 1

[7] D.H. Li and M. Fukushima, *A derivative-free line search and global convergence of Broyden-like method for nonlinear equations*, Optim. Meth. Softw, 13(3):583–599, 2000. 1

[8] W.Y. Cheng, *A PRP type method for systems of monotone equations*, Math. Comput. Model., 50(1-2):15–20, 2009. 1

[9] B.T. Polyak, *The conjugate gradient methods in extreme problems*, USSR Comput. Math. Math. Phys., 9(4):94–112, 1969. 1

[10] Q.R. Yan, X.Z. Peng, and D.H. Li, *A globally convergent derivative-free method for solving large-scale nonlinear monotone equations*, J. Comput. Appl. Math., 234(3):649–657, 2010. 1

[11] Y.H. Xiao and H. Zhu, *A conjugate gradient method to solve convex constrained monotone equations with applications in compressive sensing*, J. Math. Anal. Appl., 405(1):310–319, 2013. 1

[12] M. Li, *A derivative-free PRP method for solving large-scale nonlinear systems of equations and its global convergence*, Optim. Meth. Softw, 29(3):503–514, 2014. 1

[13] M. Rivaie, M. Mamat, L.W. June, and I. Mohd, *A new class of nonlinear conjugate gradient coefficients with global convergence properties*, Appl. Math. Comput., 218(22):11323–11332, 2012. 1

[14] W. LaCrul and M. Raydan, *Nonmonotone spectral methods for large-scale nonlinear systems*, Optim. Methods Softw., 18(5):583–599, 2003. 1

[15] Q. Li and D. Li, *A class of derivative-free methods for large-scale nonlinear monotone equations*, SIMA J. Numer. Anal., 31(4):1625–1635, 2011. 1

[16] M. Ahookhosh, K. Amini, and S. Bahrami, *Two derivative-free projection approaches for systems of large-scale nonlinear monotone equations*, Numer. Algorithms, 64(1):21–42, 2013. 1, 4, 4

[17] X.W. Fang, *A class of new derivative-free gradient type methods for large-scale nonlinear systems of monotone equations*, J. Inequalities Appl., 93:1–13, 2020. 1, 2, 3, 4, 4, 4

[18] J.K. Liu and Y.M. Feng, *A derivative-free iterative methods for nonlinear systems of monotone equations with convex constraints*, Numer. Algorithms, 82(1):245–262, 2019. 1

[19] M.V. Solodov and B.F. Svaiter, *A globally convergent inexact Newton method for systems of monotone equations*, in M. Fukushima and L. Qi (eds.), Reformulation: Nonsmooth, Piecewise Smooth, Semismooth and Smoothing Methods, Kluwer Academic Publishers, 22:355–369, 1999. 1, 2, 3

[20] L. Zhang and W.J. Zhou, *Spectral gradient projection method for solving nonlinear monotone equations*, J. Comput. Appl. Math., 196(2):478–484, 2006. 1, 2

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

[22] A.B. Abubakar and P. Kumam, *A descent Dai-Liao conjugate gradient method for nonlinear equations*, Numer. Algorithms, 81(1):197–210, 2019. 4

[23] Z. Dai and H. Zhu, *A modified Hestenes-Stiefel-type derivative-free method for large-scale nonlinear monotone equations*, Mathematics, 8(168):1–14, 2020. 4