Front. Math. China 2023, 18(4): 287–299
https://doi.org/10.3868/s140-DDD-023-0018-x

RESEARCH ARTICLE

Three-term derivative-free projection method for solving nonlinear monotone equations

Jinkui LIU, Xianglin DU

School of Mathematics and Statistics, Chongqing Three Gorges University, Chongqing 404000, China

© Higher Education Press 2023

Abstract In this paper, a three-term derivative-free projection method is proposed for solving nonlinear monotone equations. Under some appropriate conditions, the global convergence and R-linear convergence rate of the proposed method are analyzed and proved. With no need of any derivative information, the proposed method is able to solve large-scale nonlinear monotone equations. Numerical comparisons show that the proposed method is effective.

Keywords Nonlinear monotone equations, conjugate gradient method, derivative-free projection method, global convergence, R-linear convergence rate

1 Introduction

In this paper, we consider solving the nonlinear monotonic system of equations:

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

where $F: \mathbb{R}^n \to \mathbb{R}^n$ is continuous nonlinear mapping. The monotonicity of F means:

$$\langle F(x) - F(y),\ x - y\rangle \geqslant 0, \quad \forall x, y \in \mathbb{R}^n. \tag{1.2}$$

Nonlinear monotone equations are originated from many scientific and practical problems, such as first-order necessity conditions for unconstrained convex optimization problems and subproblems of generalized proximity point algorithms [9], some monotone variational inequality problems [20], chemical equilibrium problems [14], economic equilibrium problems [6], and power equations [4]. Therefore, many algorithms have been proposed to solve nonlinear monotone systems of

Translated from *Advances in Mathematics* (*China*), 2018, 47(4): 624–634
Corresponding author: Jinkui LIU, E-mail: liujinkui2006@126.com

equations, among which Newton's Method and Quasi-Newton Methods [4, 5, 11, 17, 21–23] are well known by researchers due to their excellent convergence speed. However, these algorithms are not suitable for solving large-scale non-smooth systems of equations, because they need to solve a linear system of equations using the Jacobian matrix of the objective function or an approximate Jacobian matrix at each iteration to determine the next search direction.

In recent years, in order to efficiently solve large-scale systems of equations, many researchers have promoted the nonlinear conjugate gradient method under the hyperplane projection technique [17]. For example, Cheng [3] combined the well-known PRP method [15, 16] and hyperplane projection technique to study the derivative-free PRP-type projection method, and proved the global convergence for nonlinear monotone systems of equations without requiring the objective function to be differentiable. Li et al. [12] and Ahookhosh et al. [1] further studied three derivative-free PRP-type projection algorithms. Wu et al. [18] established a derivative-free three-term HS projection algorithm based on the well-known HS conjugate gradient method [8] and hyperplane projection technique. The algorithm inherits the advantages of the HS conjugate gradient method with small storage and does not need to calculate any derivatives, thus it can solve large-scale non-smooth nonlinear monotonic systems of equations. Xiao and Zhu [19], Liu and Li [13] proposed a derivative-free CG_DESCENT type projection method based on the CG_DESCENT method [7] and hyperplane projection technique, and achieved good numerical results and proved the global convergence of the method under appropriate conditions.

Recently, Andrei [2] proposed a three-term nonlinear conjugate gradient method (denoted as TTCG method) based on the well-known CG_DESCENT method. This method is a modified memoryless BFGS algorithm whose search direction satisfies the descent and D-L conjugacy conditions. Numerous numerical experiments have shown that the TTCG method is more efficient than the well-known CG_DESCENT method. To solve the large-scale system of equations problem more effectively, this paper extends the TTCG method based on the hyperplane projection technique and establishes a three-term derivative-free projection algorithm. Under appropriate assumptions, we prove the global convergence and R-linear convergence rate of the algorithm.

2 Derivative-free projection method

The iterative method is one of the most used methods for solving nonlinear systems of equations problems. It generates an iterative column of points $\{x_k\}$, which is usually based on the following unconstrained optimization problem, namely

$$\min_{x\in\mathbb{R}^n} f(x), \tag{2.1}$$

where $f(x) = \frac{1}{2}\|F(x)\|^2$. Various iterative methods based on solving Problem (2.1) have been widely studied and applied in practice. However, the stable point of Problem (2.1) is not necessarily the solution of Problem (1.1). Moreover, in many practical problems, the mapping F is usually non-smooth.

To solve Problem (1.1) directly, Solodov and Svaiter [17] discussed the hyperplane projection method. Let x_k and d_k be the current iteration point and the search direction, respectively. A certain line search was used to obtain the step α_k so that the point $z_k = x_k + \alpha_k d_k$ is obtained such that:

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

If $F(\tilde{x}) = 0$ for some point $\tilde{x}$, then it follows from the monotonicity of F:

$$F^{\mathrm{T}}(z_k)(\tilde{x} - z_k) \leqslant 0.$$

Thus, the hyperplane $H_k = \left\{F^{\mathrm{T}}(z_k)(x - z_k) = 0\right\}$ strictly separates the current iteration point x_k from the solution of the nonlinear monotonic system of equations. Using the hyperplane, the next iteration point x_{k+1} is obtained:

$$x_{k+1} = x_k - \frac{F^{\mathrm{T}}(z_k)(x_k - z_k)}{\|F(z_k)\|^2} F(z_k). \tag{2.2}$$

The following is a three-term derivative-free projection algorithm for solving a nonlinear monotonic system of equations based on the TTCG method and the hyperplane projection method.

Algorithm 2.1
Step 0: Given the initial point $x_0 \in \mathbb{R}^n$ and the constant $\sigma \in (0,1)$, $t \geqslant 0$, $\beta > 0$. If $\|F_0\| = 0$, stop.
Step 1: Set $d_0 \in -F_0$, put $k := 0$.
Step 2: Calculate the step $\alpha_k = \beta^{m_k}$ such that m_k satisfies the following equation for the smallest non-negative integer *m:*

$$-F^{\mathrm{T}}(x_k + \alpha_k d_k)\, d_k \geqslant \sigma \alpha_k \|F(x_k + \alpha_k d_k)\| \cdot \|d_k\|^2. \tag{2.3}$$

Step 3: Let $z_k = x_k + \alpha_k d_k$. If $\|F(z_k)\| = 0$, stop. Otherwise, use (2.2) to solve for x_{k+1}.
Step 4: If $\|F_{k+1}\| = 0$, stop. Otherwise, calculate the search direction:

$$d_{k+1} = -F_{k+1} + \beta_{k+1} d_k + \theta_{k+1}(d_k + y_k), \tag{2.4}$$

where $\beta_{k+1} = \frac{1}{d_k^{\mathrm{T}}\omega_k}\left(y_k - t d_k \frac{\|y_k\|^2}{d_k^{\mathrm{T}}\omega_k}\right)^{\mathrm{T}} F_{k+1}$, $\theta_{k+1} = -\frac{F_{k+1}^{\mathrm{T}} d_k}{d_k^{\mathrm{T}}\omega_k}$, $y_k = -F_{k+1} - F_k$, $\omega_k = y_k + t_k d_k$, $t_k = 1 + \max\left\{0, \frac{d_k^{\mathrm{T}} y_k}{d_k^{\mathrm{T}} d_k}\right\}$.
Step 5: Set $k := k + 1$, turn to Step 2.

The following theorem shows that Algorithm 2.1 is a descending iterative algorithm when F is the gradient of the real-valued function $f : \mathbb{R}^n \to \mathbb{R}$.

Theorem 2.1 *Let the sequence $\{F_k\}$ and $\{d_k\}$ be generated by Algorithm* 2.1*,* *then*

$$F_k^{\mathrm{T}} d_k \leqslant -\|F_k\|^2, \quad \forall k \geqslant 0. \tag{2.5}$$

Proof According to the definition of parameter ω_k, it can be seen that

$$d_k^{\mathrm{T}} \omega_k = d_k^{\mathrm{T}} y_k + t_k \|d_k\|^2 = d_k^{\mathrm{T}} y_k + \|d_k\|^2 + \max\left\{0, -d_k^{\mathrm{T}} y_k\right\} \geqslant \|d_k\|^2. \tag{2.6}$$

Multiply both ends of (2.4) by F_{k+1}^{T} and use (2.6) to obtain

$$\begin{aligned} F_{k+1}^{\mathrm{T}} d_{k-1} &= -\|F_{k+1}\|^2 + \beta_{k+1} F_{k+1}^{\mathrm{T}} d_k + \theta_{k+1}\left(F_{k+1}^{\mathrm{T}} d_k + F_{k+1}^{\mathrm{T}} y_k\right) \\ &= -\|F_{k+1}\|^2 - \left(1 + \frac{t\|y_{\mathrm{k}}\|^2}{d_k^{\mathrm{T}} \omega_k}\right) \frac{\left(F_{k+1}^{\mathrm{T}} d_k\right)^2}{d_k^{\mathrm{T}} \omega_k} \\ &\leqslant -\|F_{k+1}\|^2. \end{aligned}$$

Known that $d_0 = -F_0$, then $F_k^{\mathrm{T}} d_0 = -\|F_0\|^2$. The conclusion is proved.

Note 2.1 According (2.5) and (2.6), we know that

$$d_k^{\mathrm{T}} \omega_k \geqslant \|F_k\|^2.$$

Since that, before Algorithm 2.1 stops, the parameter ω_k and θ_k is always meaningful.

3 Global convergences

To prove the global convergence of Algorithm 2.1, assume that the mapping F satisfies:

(H) Let the mapping F be Lipschitz continuous, i.e., there exists a constant $L > 0$ such that

$$\|F(x) - F(y)\| \leqslant L\|x - y\|, \quad \forall x, y \in \mathbb{R}^n. \tag{3.1}$$

Lemma 3.1 *Let the mapping F satisfy* (H). *The sequence $\{F_k\}$ and $\{d_k\}$ are generated by Algorithm* 2.1, *then the step factor α_k satisfies*

$$\alpha_k \geqslant \frac{\beta}{L + \sigma\left\|F\left(x_k + \beta^{-1}\alpha_k d_k\right)\right\|} \cdot \frac{\|F_k\|^2}{\|d_k\|^2}. \tag{3.2}$$

Proof If $\alpha_k \neq 1$, it follows from the definition of α_k that $\beta^{-1}\alpha_k$ does not satisfy the line search condition (2.3). Thus,

$$-\left(F\left(x_k + \beta^{-1}\alpha_k d_k\right), d_k\right) < \sigma\beta^{-1}\alpha_k \left\|F\left(x_k + \beta^{-1}\alpha_k d_k\right)\right\| \cdot \|d_k\|^2.$$

According to (2.5) and (3.1), it follows that

$$\|F_k\|^2 \leqslant -F_k^{\mathrm{T}} d_k = \left[F\left(x_k+\beta^{-1}\alpha_k d_k\right) - F\left(x_k\right)\right]^{\mathrm{T}} d_k - F\left(x_k+\beta^{-1}\alpha_k d_k\right)^{\mathrm{T}} d_k$$
$$\leqslant L\beta^{-1}\alpha_k \|d_k\|^2 + \sigma\beta^{-1}\alpha_k \left\|F\left(x_k+\beta^{-1}\alpha_k d_k\right)\right\| \cdot \|d_k\|^2.$$

So, it can be concluded that

$$\alpha_k \geqslant \frac{\beta}{L+\sigma\left\|F\left(x_k+\beta^{-1}\alpha_k d_k\right)\right\|} \cdot \frac{\|F_k\|^2}{\|d_k\|^2}.$$

The lemma is proven.

The following lemma shows that sequence $\{\|x_k - \tilde{x}\|\}$ is descending and convergent, where sequence $\{x_k\}$ is generated by Algorithm 2.1 and point $\tilde{x}$ is an arbitrary solution to Problem (1.1). The proof process of this lemma can be found in Reference [17].

Lemma 3.2 *Assuming that mapping F satisfies* (H) *and sequence $\{x_k\}$ and $\{z_k\}$ are generated by Algorithm* 2.1, *then for any solution $\tilde{x}$ to Problem* (1.1), *there is*

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

Especially, the sequence $\{x_k\}$ is bounded and

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

Note 3.1 According to (2.2) and (2.3), we know that

$$\|x_{k+1} - x_k\| = \frac{\alpha_k \left|F^{\mathrm{T}}\left(z_k\right) d_k\right|}{\left\|F\left(z_k\right)\right\|^2} \cdot \left\|F\left(z_k\right)\right\| \geqslant \sigma \|\alpha_k d_k\|^2.$$

According to (3.4), we know that

$$\lim_{k\to\infty} \|\alpha_k d_k\| = 0. \tag{3.5}$$

Lemma 3.3 *Assuming that the mapping F satisfies* (H), *the sequences $\{d_k\}$ and $\{F_k\}$ are generated by Algorithm* 2.1, *then there exists a positive real number M such that*

$$\|d_k\| \leqslant M, \quad \forall k \geqslant 0. \tag{3.6}$$

Proof For a certain solution $\tilde{x}$ of Problem (1.1) satisfies

$$\|F_k\| = \|F\left(x_k\right) - F\left(\tilde{x}\right)\| \leqslant L\left\|x_k - \tilde{x}\right\|.$$

Then, applying the boundedness of sequence $\{x_k\}$, we can see that there is a normal number μ, which makes

$$\|F_k\| \leqslant \mu, \quad \forall k \geqslant 0. \tag{3.7}$$

By using (2.2) and (3.1), as well as the Cauchy-Schwartz inequality and the definition of step α_k, it can be obtained that

$$\|y_k\| \leqslant L\left\|x_{k+1}-x_k\right\| \leqslant \frac{L\|F(z_k)\|^2 \cdot \|x_k-z_k\|}{\|F(z_k)\|^2} = L\left\|\alpha_k d_k\right\| \leqslant L\beta\left\|d_k\right\|. \tag{3.8}$$

According to (2.6), (3.7), (3.8) and Cauchy-Schwartz inequality, it can be obtained that

$$\begin{aligned}
|\beta_{k+1}| &\leqslant \frac{\|F_{k+1}\| \cdot \|y_k\|}{d_k^{\mathrm{T}}\omega_k} + \frac{t\|y_k\|^2 \cdot \|F_{k+1}\| \cdot \|d_k\|}{(d_k^{\mathrm{T}}\omega_k)^2} \\
&\leqslant \frac{L\beta\left\|F_{k+1}\right\| \cdot \|d_k\|}{\|d_k\|^2} + \frac{tL^2\beta^2\|d_k\|^2 \cdot \|F_{k+1}\| \cdot \|d_k\|}{\|d_k\|^4} \\
&\leqslant \frac{L\beta\mu}{\|d_k\|} + \frac{t\beta^2L^2\mu}{\|d_k\|}.
\end{aligned}$$

$$|\theta_{k+1}| \leqslant \frac{\|F_{k+1}\| \cdot \|d_k\|}{\|d_k\|^2} \leqslant \frac{\mu}{\|d_k\|}.$$

Based on the above two inequalities and (2.4), (3.8), it can be concluded that

$$\begin{aligned}
\|d_{k+1}\| &\leqslant \|F_{k+1}\| + |\beta_{k+1}| \cdot \|d_k\| + |\theta_{k+1}|\left(\|d_k\| + \|y_k\|\right) \\
&\leqslant \mu + \left(\beta L\mu + t\beta^2L^2\mu\right) + \mu\left(1+L\beta\right).
\end{aligned}$$

Since $d_0=-F_0$, take $M=\mu+(\beta L\mu+t\beta^2L^2\mu)+\mu\left(1+L\beta\right)$, the conclusion is proved. □

Theorem 3.1 *If the mapping F satisfies* (H), *the sequences $\{d_k\}$ and $\{F_k\}$ are generated by Algorithm* 2.1, *then*

$$\lim_{k\to\infty}\inf\|F_k\| = 0. \tag{3.9}$$

Proof Assuming (3.9) doesn't hold, there exists a normal number ε such that

$$\|F_k\| \geqslant \varepsilon, \quad \forall k \geqslant 0. \tag{3.10}$$

From (3.2), (3.6) and (3.7), it can be found that

$$\alpha_k\left\|d_k\right\| \geqslant \frac{\beta\|F_k\|^2}{\left(L+\sigma\left\|F\left(x_k+\beta^{-1}\alpha_k\right)\right\| \cdot \|d_k\|\right)} \geqslant \frac{\beta\varepsilon^2}{\left(L+\sigma\mu\right)M} > 0.$$

Obviously, it is contradictory with (3.5). Therefore, the hypothesis does not hold and the original proposition holds.

4 *R*-Linear convergence rate

To further prove the R-linear convergence rate of Algorithm 2.1, it is necessary to assume that the mapping F satisfies:

(G) Let $\forall \tilde{x} \in \Omega$, there exist positive constants $\rho \in (0,1)$ and $\nu > 0$ such that

$$\rho \operatorname{dist}(x, \Omega) \leqslant \|F_k\|^2, \quad \forall x \in N(\tilde{x}, \nu), \tag{4.1}$$

where Ω is the set of solutions of Problem (1.1), $N(\tilde{x}, \nu) = \{x \in \mathbb{R}^n \,|\, \|x - \tilde{x}\| \leqslant \nu\}$, $\operatorname{dist}(x, \Omega)$ denotes the distance of point x from the solution set Ω.

Theorem 4.1 *Suppose that the mapping F satisfies* (H) *and* (G), *the sequence $\{x_k\}$ is generated by Algorithm* 2.1, *then the sequence $\{dist(x_k, \Omega)\}$ is Q-linearly convergent to* 0. *Thus, the sequence $\{x_k\}$ is R-linearly convergent.*

Proof Let $\omega_k = \arg\min\{\|x_k - \omega\| \mid \omega \in \Omega\}$. we can know ω_k is the solution closest to point x_k in solution set Ω, i.e.,

$$\operatorname{dist}(x_{k+1}, \Omega) = \|x_k - \omega_k\|. \tag{4.2}$$

By using (2.5) and Cauchy-Schwartz inequality, it can be obtained that

$$\|d_k\| \geqslant \|F_k\|. \tag{4.3}$$

Known that $\omega_k \in \Omega$, using (3.3), (4.1)–(4.3) and Note 3.1, we can get

$$\begin{aligned}
\operatorname{dist}(x_{k+1}, \Omega)^2 &\leqslant \|x_{k+1} - \omega_k\|^2 \leqslant \|x_k - \omega_k\|^2 - \|x_{k+1} - x_k\|^2 \\
&= dist(x_k, \Omega)^2 - \|x_{k+1} - x_k\|^2 \\
&\leqslant \operatorname{dist}(x_k, \Omega)^2 - \sigma^2 \|\alpha_k d_k\|^2 \\
&\leqslant \operatorname{dist}(x_k, \Omega)^2 - \sigma^2 \alpha_k^4 \|F_k\|^4 \\
&\leqslant \left(1 - \sigma^2 \alpha_k^4 \rho^2\right) \operatorname{dist}(x_k, \Omega)^2.
\end{aligned}$$

This shows that the sequence $\{\operatorname{dist}(x_k, \Omega)\}$ is Q-linearly convergent to 0. Since $1 - \sigma^2 \alpha_k^4 \rho^2 \in (0,1)$, the sequence $\{x_k\}$ is R-linearly convergent.

5 Numerical tests

Algorithm 2.1 is tested numerically below and compared numerically with the MPRP-based method [12] and the MHS-based method [18]. The program code is written using MATLAB 7.0, running on an Intel Core i5-4590 processor (quad-core, 3.30 GHz), 8.0 GB of internal memory, and the Windows 7 operating system. The selected test problems were taken as follows, from the literature [22, 3, 10], respectively.

Problem 5.1 [22] F is defined by the following equation:

$$F(x) = A(x) + g(x) - 1,$$

where $g(x) = (\mathrm{e}^{x_1}, \mathrm{e}^{x_2}, \ldots, \mathrm{e}^{x_n})^{\mathrm{T}}$ and

$$A=\begin{pmatrix} 2 & -1 & & & \\ -1 & 2 & -1 & & \\ & \ddots & \ddots & \ddots & \\ & & \ddots & \ddots & -1 \\ & & & -1 & 2 \end{pmatrix}.$$

Problem 5.2 [3] F is defined by the following equation:

$$F(x)=A(x)-1,$$

where

$$A=\begin{pmatrix} 2.5 & 1 & & & \\ 1 & 2.5 & 1 & & \\ & \ddots & \ddots & \ddots & \\ & & \ddots & \ddots & 1 \\ & & & 1 & 2.5 \end{pmatrix}.$$

Problem 5.3 [10] F is defined by the following equation:

$$F(x)=(F_1(x),F_2(x),\ldots,F_n(x))^{\mathrm{T}},$$

where

$$\begin{aligned} &F_1(x)=x_1+\mathrm{e}^{\cos\left(\frac{x_1+x_2}{n+1}\right)},\\ &F_i(x)=x_i+\mathrm{e}^{\cos\left(\frac{x_{i-1}+x_i+x_{i+1}}{n+1}\right)},\quad i=2,3,\ldots,n-1.\\ &F_n(x)=x_n+\mathrm{e}^{\cos\left(\frac{x_{n-1}+x_n}{n+1}\right)}. \end{aligned}$$

The parameter value in Algorithm 2.1 is $\sigma=0.01$, $t=2$, $\beta=0.5$. The termination conditions for all programs are $\|F(x_k)\|\leqslant 1.0\times10^{-5}$ or $\|F(z_k)\|\leqslant 1.0\times10^{-5}$.

In numerical tests, the rand function in MATLAB software is used to randomly generate a fixed initial point within $(-1,0),(0,1),(-1,1),(-2,0),(0,2),(-5,5),(-10,10)$, and calculate three test problems. The numerical results are shown in Tables 1–7. In Tables 1–7, "Dim" represents the dimension of the test problem, "NI" represents the number of iterations, and "NF" represents the number of calculations for F, "CPU " represents CPU time (in seconds).

Tables 1–7 indicate that Algorithm 2.1, PRP-based method, and HS-based method can successfully solve the given test problem starting from randomly generated initial points. For Problems 5.1 and 5.2, the numerical effects of the three test methods are comparable; However, for Problem 5.3, the performance of Algorithm 2.1 is significantly better than that of PRP-based and HS-based methods.

Table 1 Calculation results of Algorithm 2.1, MPRP-based method and MHS-based method, with initial points in the interval $(-1, 0)$

	Dim	Algorithm 2.1			PRP-based method			HS-based method		
		NI	NF	CPU	NI	NF	CPU	NI	NF	CPU
Problem 5.1	5000	32	96	0.06	39	113	0.08	31	110	0.03
	7000	32	96	0.09	39	113	0.08	31	110	0.05
	9000	32	96	0.09	40	116	0.09	32	113	0.06
	11000	33	99	0.09	40	116	0.13	32	113	0.08
	13000	33	99	0.14	33	92	0.16	32	113	0.08
	15000	33	99	0.13	33	92	0.14	32	113	0.11
Problem 5.2	5000	41	115	0.06	37	96	0.05	40	132	0.03
	7000	43	121	0.08	42	110	0.09	49	159	0.07
	9000	39	105	0.08	38	97	0.08	47	159	0.08
	11000	39	105	0.6	38	97	0.09	52	173	0.09
	13000	39	105	0.09	39	101	0.08	61	204	0.11
	15000	40	108	0.1	39	101	0.11	47	156	0.11
Problem 5.3	5000	10	23	0.06	26	54	0.06	20	42	0.03
	7000	11	26	0.05	26	55	0.09	20	43	0.04
	9000	11	26	0.08	27	57	0.11	21	46	0.06
	11000	12	30	0.06	27	57	0.13	21	47	0.07
	13000	12	30	0.08	27	58	0.14	21	48	0.08
	15000	13	34	0.09	27	58	0.14	22	50	0.10

Table 2 Calculation results of Algorithm 2.1, MPRP-based method and MHS-based method, with initial points in the interval $(-2, 1)$

	Dim	Algorithm 2.1			PRP-based method			HS-based method		
		NI	NF	CPU	NI	NF	CPU	NI	NF	CPU
Problem 5.1	5000	30	92	0.05	37	109	0.06	29	102	0.05
	7000	30	92	0.08	38	112	0.09	29	102	0.06
	9000	30	92	0.09	38	112	0.09	29	102	0.05
	11000	31	95	0.08	33	96	0.11	29	102	0.09
	13000	31	95	0.11	33	96	0.14	29	102	0.09
	15000	31	95	0.11	33	96	0.14	30	106	0.09
Problem 5.2	5000	34	83	0.09	46	118	0.08	51	170	0.05
	7000	39	101	0.05	42	107	0.08	46	150	0.07
	9000	39	101	0.06	45	117	0.08	62	209	0.10
	11000	39	100	0.08	44	114	0.08	49	160	0.08
	13000	40	103	0.11	44	113	0.13	45	149	0.10
	15000	41	109	0.09	41	109	0.11	42	139	0.10
Problem 5.3	5000	9	19	0.05	24	49	0.08	19	39	0.03
	7000	9	19	0.05	25	51	0.11	19	40	0.07
	9000	9	19	0.08	25	52	0.11	19	40	0.04
	11000	10	23	0.05	26	54	0.13	20	42	0.05
	13000	11	26	0.08	26	54	0.14	20	42	0.06
	15000	11	26	0.11	26	55	0.17	20	43	0.09

Table 3 Calculation results of Algorithm 2.1, MPRP-based method and MHS-based method, with initial points in the interval (−1, 1)

	Dim	Algorithm 2.1			PRP-based method			HS-based method		
		NI	NF	CPU	NI	NF	CPU	NI	NF	CPU
Problem 5.1	5000	31	97	0.06	39	117	0.07	30	109	0.07
	7000	32	100	0.08	39	117	0.13	33	121	0.08
	9000	33	103	0.09	40	120	0.11	31	113	0.07
	11000	33	103	0.10	40	120	0.12	31	113	0.08
	13000	33	103	0.10	40	120	0.14	32	116	0.11
	15000	33	103	0.13	41	123	0.18	32	116	0.10
Problem 5.2	5000	39	98	0.06	38	97	0.07	51	176	0.05
	7000	38	96	0.06	47	125	0.07	44	145	0.05
	9000	38	96	0.07	43	110	0.06	49	166	0.07
	11000	39	98	0.09	43	110	0.13	50	170	0.09
	13000	39	98	0.11	43	110	0.11	57	191	0.12
	15000	39	98	0.11	44	113	0.14	52	169	0.11
Problem 5.3	5000	9	19	0.04	25	51	0.11	19	40	0.04
	7000	10	23	0.05	26	54	0.13	20	42	0.05
	9000	11	26	0.06	26	54	0.11	20	43	0.05
	11000	11	26	0.06	26	55	0.16	21	46	0.07
	13000	12	30	0.09	27	57	0.17	21	46	0.08
	15000	12	30	0.09	27	57	0.16	21	47	0.09

Table 4 Calculation results of Algorithm 2.1, MPRP-based method and MHS-based method, with initial points in the interval (−2, 1)

	Dim	Algorithm 2.1			PRP-based method			HS-based method		
		NI	NF	CPU	NI	NF	CPU	NI	NF	CPU
Problem 5.1	5000	32	97	0.09	40	115	0.06	56	235	0.08
	7000	38	116	0.09	34	95	0.11	56	235	0.11
	9000	38	116	0.09	34	95	0.08	41	157	0.09
	11000	38	116	0.13	34	95	0.09	42	161	0.12
	13000	38	116	0.13	35	99	0.11	39	147	0.14
	15000	38	116	0.16	35	99	0.14	39	147	0.14
Problem 5.2	5000	43	118	0.06	43	111	0.06	50	170	0.05
	7000	43	117	0.08	44	113	0.09	49	165	0.06
	9000	41	111	0.09	42	109	0.08	47	158	0.07
	11000	43	117	0.09	45	117	0.12	51	172	0.09
	13000	43	117	0.11	44	115	0.09	47	156	0.09
	15000	43	117	0.14	49	129	0.12	43	141	0.10
Problem 5.3	5000	11	26	0.05	26	54	0.06	20	43	0.04
	7000	12	30	0.05	27	57	0.11	21	46	0.05
	9000	12	30	0.13	27	58	0.09	21	48	0.06
	11000	13	34	0.08	27	58	0.11	22	50	0.08
	13000	13	34	0.08	28	62	0.14	22	51	0.09
	15000	14	39	0.09	29	65	0.17	22	52	0.10

Table 5 Calculation results of Algorithm 2.1, MPRP-based method and MHS-based method, with initial points in the interval (0, 2)

	Dim	Algorithm 2.1			PRP-based method			HS-based method		
		NI	NF	CPU	NI	NF	CPU	NI	NF	CPU
Problem 5.1	5000	34	106	0.08	32	94	0.06	33	121	0.03
	7000	34	106	0.06	32	94	0.09	34	125	0.07
	9000	34	106	0.11	32	94	0.09	34	125	0.08
	11000	35	109	0.08	33	97	0.12	34	125	0.09
	13000	35	109	0.13	33	95	0.13	34	125	0.11
	15000	35	109	0.13	33	95	0.17	34	125	0.12
Problem 5.2	5000	42	115	0.06	44	116	0.05	47	159	0.03
	7000	42	114	0.08	47	124	0.06	47	159	0.06
	9000	42	115	0.09	43	110	0.08	53	182	0.08
	11000	43	116	0.09	43	112	0.11	48	163	0.08
	13000	43	116	0.17	44	115	0.12	48	164	0.10
	15000	44	120	0.12	44	115	0.14	60	209	0.13
Problem 5.3	5000	9	19	0.06	24	49	0.06	18	37	0.04
	7000	9	19	0.06	24	49	0.08	19	39	0.03
	9000	9	19	0.08	25	51	0.11	19	40	0.05
	11000	9	19	0.06	25	51	0.17	19	40	0.06
	13000	9	19	0.06	25	52	0.12	19	40	0.07
	15000	10	23	0.08	25	52	0.16	19	40	0.08

Table 6 Calculation results of Algorithm 2.1, MPRP-based method and MHS-based method, with initial points in the interval (−5, 5)

	Dim	Algorithm 2.1			PRP-based method			HS-based method		
		NI	NF	CPU	NI	NF	CPU	NI	NF	CPU
Problem 5.1	5000	37	118	0.08	39	118	0.09	43	172	0.07
	7000	41	135	0.14	39	118	0.11	45	181	0.09
	9000	39	130	0.13	39	120	0.11	46	188	0.12
	11000	44	147	0.14	41	127	0.13	45	184	0.14
	13000	42	144	0.16	41	127	0.19	45	184	0.16
	15000	44	151	0.17	42	132	0.17	45	186	0.18
Problem 5.2	5000	42	108	0.06	47	119	0.06	53	184	0.05
	7000	44	116	0.08	47	123	0.09	59	206	0.07
	9000	44	116	0.06	48	126	0.09	46	156	0.06
	11000	46	125	0.11	47	124	0.09	59	206	0.10
	13000	46	125	0.09	47	124	0.11	64	226	0.13
	15000	48	135	0.13	51	137	0.16	60	216	0.14
Problem 5.3	5000	11	26	0.13	26	55	0.13	20	44	0.04
	7000	12	30	0.06	27	57	0.11	21	46	0.07
	9000	12	30	0.06	27	58	0.13	22	50	0.06
	11000	13	34	0.11	28	62	0.14	22	50	0.08
	13000	14	39	0.09	28	62	0.16	22	52	0.09
	15000	14	39	0.11	29	66	0.19	23	56	0.11

Table 7 Calculation results of Algorithm 2.1, MPRP-based method and MHS-based method, with initial points in the interval (−10, 10)

	Dim	Algorithm 2.1			PRP-based method			HS-based method		
		NI	NF	CPU	NI	NF	CPU	NI	NF	CPU
Problem 5.1	5000	47	182	0.09	47	167	0.13	80	386	0.13
	7000	47	183	0.13	49	177	0.16	207	1349	0.55
	9000	49	197	0.17	49	183	0.16	211	1375	0.71
	11000	50	202	0.19	51	189	0.17	193	1205	0.76
	13000	53	221	0.23	52	193	0.23	173	1064	0.78
	15000	51	214	0.27	52	200	0.27	89	463	0.45
Problem 5.2	5000	49	138	0.09	57	155	0.08	69	255	0.06
	7000	51	146	0.09	51	139	0.09	62	229	0.08
	9000	51	149	0.09	57	156	0.14	83	303	0.11
	11000	50	146	0.11	61	171	0.17	89	323	0.16
	13000	50	147	0.14	57	162	0.14	66	246	0.13
	15000	51	151	0.12	58	168	0.16	81	303	0.19
Problem 5.3	5000	13	34	0.05	28	62	0.11	22	51	0.04
	7000	14	40	0.09	30	70	0.09	23	57	0.07
	9000	16	48	0.06	31	73	0.12	24	62	0.08
	11000	17	54	0.08	31	74	0.16	25	66	0.10
	13000	18	58	0.14	32	78	0.19	26	70	0.12
	15000	18	60	0.16	33	81	0.20	27	76	0.14

References

1. Ahookhosh M, Amini K, Bahrami S. Two derivative-free projection approaches for systems of largescale nonlinear monotone equations. Numer Algorithms, 2013, 64(1): 21–42
2. Andrei N. On three-term conjugate gradient algorithms for unconstrained optimization. Appl Math Comput, 2013, 219: 6316–6317
3. Cheng W Y. A PRP type method for systems of monotone equations. Math Comput Modelling, 2009, 50: 15–20
4. Dennis J E, More J J. A characterization of superlinear convergence and its application to quasi-Newton methods. Math Comp, 1974, 28(126): 549–560
5. Dennis J E, More J J. Quasi-Newton method, motivation and theory. SIAM Rev, 1997, 19(1): 46–89
6. Dirkse S P, Ferris M C. MCPLIB: A collection of nonlinear mixed complementarity problems. Optim Methods Softw, 2002, 5(4): 319–345
7. Hager W W, Zhang H. A new conjugate gradient method with guaranteed descent and an efficient line search. SIAM J Optim, 2005, 16(1): 170–192
8. Hestens M R, Stiefel E L. Methods of conjugate gradients for solving linear systems. J Research Nat Bur Standards, 1952, 49(6): 409–436
9. Iusem A N, Solodov M V. Newton-type methods with generalized distances for constrained optimization. Optimization, 1997, 41(3): 257–278
10. La Cruz M, Martinez J M, Raydan M. Spectral residual method without gradient information for solving large-scale nonlinear systems of equations. Math Comp, 2006, 75(255): 1429–1448
11. Li D H, Fukushima M. A global and superlinear convergent Gauss-Newton-based BFGS method for symmetric nonlinear equations. SIAM J Numer Anal, 1999,

37(1): 152–172
12. Li Q N, Li D H. A class of derivative-free methods for large-scale nonlinear monotone equations. IMA J Numer Anal, 2011, 31(4): 1625–1635
13. Liu J K, Li S J. A projection method for convex constrained monotone nonlinear equations with applications. Comput Math Appl, 2015, 70(10): 2442–2453
14. Meintjes K, Morgan A P. A methodology for solving chemical equilibrium systems. Appl Math Comput, 1987, 22(4): 333–361
15. Polyak B T. The conjugate gradient method in extremal problems. USSR Comput Math and Math Phys, 1969, 9(4): 94–112
16. Polak E, Ribière G. Note sur la convergence de méthodes de directions conjuguées. Rev Francaise Informat Recherche Opérationnelle, 1969, 3(16): 35–43 (in French)
17. Solodov M V, Svaiter B F. A globally convergent inexact Newton method for systems of monotone equations. In: Reformulation: Non-smooth, Piecewise Smooth, Semismooth and Smoothing Methods. Dordrecht: Springer, 1999, 355–369
18. Wu X Y, Sai N E Z, Zhang H L. On three-term HS projection algorithm for solving nonlinear monotone equations. Southwest China Normal Univ Natur Sci Ed, 2016, 41(5): 41–47
19. Xiao Y H, Zhu H. A conjugate gradient method to solve convex constrained monotone equations with applications in compressive sensing. J Math Anal Appl, 2013, 405(1): 310–319
20. Zhao Y B, Li D H. Monotonicity of fixed point and normal mapping associated with variational inequality and its application. SIAM J Optim, 2000, 11(4): 962–973
21. Zhou G, Toh K C. Superline convergence of a Newton-type algorithm for monotone equations. J Optim Theory Appl, 2005, 125(1): 205–221
22. Zhou W J, Li D H. Limited memory BFGS method for nonlinear monotone equations. J Comput Math, 2007, 25(1): 89–96
23. Zhou W J, Li D H. A globally convergent BFGS method for nonlinear monotone equations without any merit functions. Math Comp, 2008, 77(264): 2231–2240