Mathematical Modelling and Analysis
Volume 24, Issue 4, 550–563, 2019
https://doi.org/10.3846/mma.2019.033

http://mma.vgtu.lt
ISSN: 1392-6292
eISSN: 1648-3510

A New Three-Term Conjugate Gradient-Based Projection Method for Solving Large-Scale Nonlinear Monotone Equations

Mompati Koorapetse and Professor Kaelo

University of Botswana
Department of Mathematics, Private Bag UB00704, Gaborone
E-mail(*corresp.*): `kaelop@mopipi.ub.bw`
E-mail: `sphetto77@gmail.com`

Received January 22, 2019; revised September 1, 2019; accepted September 2, 2019

Abstract. A new three-term conjugate gradient-based projection method is presented in this paper for solving large-scale nonlinear monotone equations. This method is derivative-free and it is suitable for solving large-scale nonlinear monotone equations due to its lower storage requirements. The method satisfies the sufficient descent condition $F_k^T d_k \leq -\tau \|F_k\|^2$, where $\tau > 0$ is a constant, and its global convergence is also established. Numerical results show that the method is efficient and promising.

Keywords: nonlinear monotone equations, derivative-free, global convergence.

AMS Subject Classification: 90C06; 90C30; 90C56; 65K05; 65K10.

1 Introduction

Conjugate gradient-based projection methods are a class of methods suited for solving large-scale nonlinear equations

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

where $F : \mathbb{R}^n \to \mathbb{R}^n$ is a continuous and monotone function. A function F is said to be monotone if it satisfies

$$(F(x) - F(y))^T (x - y) \geq 0, \quad \forall x, y \in \mathbb{R}^n.$$

Nonlinear monotone equations arise in many practical applications, for example, as chemical equilibrium systems [13], economic equilibrium problems [5], and signal and image recovery problems [8]. Some mathematical problems can also be transformed into finding the solution of problem (1.1), such as variational inequality problems. This wide application has seen a number of researchers study iterative schemes (methods) for solving monotone equations over the years.

Conjugate gradient methods [3,9,17,22] are very efficient methods for solving large-scale unconstrained optimization problems mainly due to their simplicity and low storage requirements. This has over the years influenced researchers to propose conjugate gradient-based projection methods by combining conjugate gradient methods with the hyperplane projection method [16] to solve nonlinear monotone equations.

Conjugate gradient-based projection methods are iterative methods that generate the next iterate x_{k+1}, given x_k, by

$$x_{k+1} = x_k - \frac{F(z_k)^T(x_k - z_k)}{\|F(z_k)\|^2} F(z_k), \tag{1.2}$$

where $z_k = x_k + \alpha_k d_k$, $\|\cdot\|$ denotes the Euclidean norm and

$$d_k = \begin{cases} -F_k, & \text{if } k = 0, \\ -F_k + \beta_k d_{k-1}, & \text{if } k \geq 1, \end{cases}$$

with β_k a parameter such that

$$F_k^T d_k \leq -\tau \|F_k\|^2, \ \tau > 0, \tag{1.3}$$

$F_k = F(x_k)$ and α_k is a step length. One such method is that by Papp and Rapajić [14] who proposed a derivative-free projection method

$$d_k = \begin{cases} -F_k, & \text{if } k = 0, \\ -F_k + \beta_k^{FR} w_{k-1} - \theta_k F_k, & \text{if } k \geq 1, \end{cases}$$

where

$$\beta_k^{FR} = \frac{\|F_k\|^2}{\|F_{k-1}\|^2}, \quad \theta_k = \frac{\|F_k\|^2 \|w_{k-1}\|^2}{\|F_{k-1}\|^4} \quad \text{and} \quad w_{k-1} = z_k - x_k = \alpha_k d_k.$$

The global convergence of this method was established using the line search

$$-F(z_k)^T d_k \geq \sigma \alpha_k \|F(z_k)\| \|d_k\|^2,$$

with $\sigma > 0$ being a constant.

Another method is that by Sun and Liu [18] which proposes the search direction

$$d_k = \begin{cases} -F_k, & \text{if } k = 0, \\ -\left(1 + \beta_k \frac{F_k^T d_{k-1}}{\|F_k\|^2}\right) F_k + \beta_k d_{k-1}, & \text{if } k \geq 1, \end{cases}$$

where $\beta_k = t\|F_k\|/\|d_{k-1}\|$, for some positive constant t. They proved that the method is globally convergent and showed through numerical results that the method is very efficient.

More recently, Liu and Feng [10] proposed a derivative-free iterative method

$$d_k = \begin{cases} -F_k, & \text{if } k = 0, \\ -\theta_k F_k + \beta_k d_{k-1}, & \text{if } k \geq 1, \end{cases}$$

where $\beta_k = \|F_k\|^2/(d_{k-1}^T w_{k-1})$, with $w_{k-1} = y_{k-1} + t d_{k-1}$, $y_{k-1} = F_k - F_{k-1}$, $t = 1 + \max\left\{0, -\frac{d_{k-1}^T y_{k-1}}{\|d_{k-1}\|^2}\right\}$ and $\theta_k = \tau - \frac{F_k^T d_{k-1}}{d_{k-1}^T w_{k-1}}$, for some positive constant τ that satisfies (1.3). This method was applied on convex constrained monotone equations. For more conjugate gradient-based projection methods, the reader is referred to [1, 2, 4, 7, 11, 12, 15, 19, 20, 21].

In this paper, motivated by the work of Zheng and Zheng [22], we propose another conjugate gradient-based projection method. This method is presented in the next section. In Section 3, we prove the global convergence of the proposed method. Numerical results follow in Section 4 and conclusion is presented in Section 5.

2 Algorithm

Recently, in solving an unconstrained optimization problem

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

where $f : \mathbb{R}^n \to \mathbb{R}$ is a continuously differentiable function, Zheng and Zheng [22] proposed two conjugate gradient methods that generate d_k by

$$d_k = \begin{cases} -g_k, & \text{if } k = 0, \\ -g_k + \beta_k d_{k-1}, & \text{if } k \geq 1, \end{cases}$$

with $g_k = \nabla f(x_k)$ being the gradient of f at x_k and the parameter β_k given by

$$\beta_k^{DHSDL} = \frac{\|g_k\|^2 - \frac{\|g_k\|}{\|g_{k-1}\|}|g_k^T g_{k-1}|}{\mu |g_k^T d_{k-1}| + d_{k-1}^T y_{k-1}} - t\frac{g_k^T s_{k-1}}{d_{k-1}^T y_{k-1}},$$

$$\beta_k^{DLSDL} = \frac{\|g_k\|^2 - \frac{\|g_k\|}{\|g_{k-1}\|}|g_k^T g_{k-1}|}{\mu |g_k^T d_{k-1}| - d_{k-1}^T g_{k-1}} - t\frac{g_k^T s_{k-1}}{d_{k-1}^T y_{k-1}},$$

where $y_{k-1} = g_k - g_{k-1}$, $\mu > 1$ and $t > 0$. This method was shown to converge globally using the strong Wolfe line search and it was also shown to perform very well numerically.

Now, motivated the definition of β_k by Zheng and Zheng [22], we propose a new conjugate gradient-based projection method for (1.1) by defining the search direction as

$$d_k = \begin{cases} -F_k, & \text{if } k = 0, \\ -F_k + \beta_k d_{k-1} - \theta_k w_{k-1}, & \text{if } k \geq 1, \end{cases} \tag{2.1}$$

where

$$\beta_k = \frac{\|F_k\|^2 - \frac{\|F_k\|}{\|F_{k-1}\|}|F_k^T F_{k-1}|}{\mu\|F_k\|\|d_{k-1}\| - F_{k-1}^T d_{k-1}}, \quad \theta_k = \frac{F_k^T w_{k-1}}{\mu\|w_{k-1}\|^2}, \tag{2.2}$$

$w_{k-1} = y_{k-1} + d_{k-1}$, $y_{k-1} = F_k - F_{k-1} + rs_{k-1}$, $s_{k-1} = x_k - x_{k-1}$, $\mu > 1$ and $r > 0$. Note here that w_{k-1} is defined as in [10, 11] except that in our case t is always taken to be $t = 1$. The specific steps of our proposed method are presented below in Algorithm 1.

Algorithm 1 [Three-term Conjugate Gradient-based Method (TCGM)].

1. Give $x_0 \in \mathbb{R}^n$, the parameters σ, κ, r, μ and $\rho \in (0,1)$. Set $k = 0$.
2. FOR $k = 0, 1, \ldots$ do
3. If $\|F_k\| = 0$, then stop. Otherwise, go to Step 4.
4. Compute d_k by (2.1)–(2.2).
5. Compute $z_k = x_k + \alpha_k d_k$ where $\alpha_k = \max\{\kappa\rho^i : i = 0, 1, 2, ...\}$ such that the inequality
$$-F(x_k + \alpha_k d_k)^T d_k \geq \sigma\alpha_k \parallel d_k \parallel^2 \tag{2.3}$$
with $\sigma > 0$ and $\kappa > 0$ being constants, is satisfied.
6. If $\|F(z_k)\| = 0$, then stop. Otherwise, compute x_{k+1} using (1.2).
7. Set $k = k + 1$ and go to Step 3.
8. ENDFOR

Throughout this paper, we assume that the following assumption holds.

Assumption 1.

(i) The function $F(\cdot)$ is monotone on $\mathbb{R}^n$.

(ii) The function $F(\cdot)$ is Lipschitz continuous on $\mathbb{R}^n$, i.e. there exists a positive constant L such that

$$\parallel F(x) - F(y) \parallel \leq L \parallel x - y \parallel, \quad \forall x, y \in \mathbb{R}^n.$$

(iii) The solution set of (1.1) is nonempty.

3 Convergence analysis

We now present the global convergence of our algorithm under Assumption 1.

Lemma 1. *Suppose that Assumption 1 holds. Let the sequence $\{x_k\}$ be generated by Algorithm 1. Then the search direction d_k satisfies the sufficient descent condition*

$$F_k^T d_k \leq -\left(1 - \frac{1}{\mu}\right)\|F_k\|^2, \quad \forall k \geq 0 \quad \text{and} \quad \mu > 1. \tag{3.1}$$

Proof. Since $d_0 = -F_0$, we have $F_0^T d_0 = -\|F_0\|^2$. From this, we get that $F_0^T d_0 \leq -\left(1 - \frac{1}{\mu}\right)\|F_0\|^2$ is satisfied for any $\mu > 1$. To prove (3.1) for $k \geq 1$, we assume $F_{k-1}^T d_{k-1} \leq -\left(1 - \frac{1}{\mu}\right)\|F_{k-1}\|^2$ and show it is true for k. Notice from the definition of β_k that we have

$$0 \leq \beta_k \leq \frac{\|F_k\|^2}{\mu\|F_k\|\|d_{k-1}\|}.$$

Now, we obtain from (2.1)–(2.2) that

$$\begin{aligned}
F_k^T d_k &= -\|F_k\|^2 + \beta_k F_k^T d_{k-1} - \theta_k F_k^T w_{k-1} \\
&= -\|F_k\|^2 + \frac{\|F_k\|^2 - \frac{\|F_k\|}{\|F_{k-1}\|}|F_k^T F_{k-1}|}{\mu\|F_k\|\|d_{k-1}\| - F_{k-1}^T d_{k-1}} F_k^T d_{k-1} - \frac{(F_k^T w_{k-1})^2}{\mu\|w_{k-1}\|^2} \\
&\leq -\|F_k\|^2 + \frac{\|F_k\|^2}{\mu\|F_k\|\|d_{k-1}\|}\|F_k\|\|d_{k-1}\| = -\left(1 - \frac{1}{\mu}\right)\|F_k\|^2.
\end{aligned}$$

Hence (3.1) is satisfied for all $k \geq 0$. □

The following lemma indicates that if the sequence $\{x_k\}$ is generated by Algorithm 1 and x^* is such that $F(x^*) = 0$, then the sequence $\{x_k - x^*\}$ is decreasing and convergent, thus the sequence $\{x_k\}$ is bounded.

Lemma 2. *Suppose Assumption 1 holds and the sequence $\{x_k\}$ is generated by Algorithm 1. For any x^* such that $F(x^*) = 0$, we have that*

$$\|x_{k+1} - x^*\|^2 \leq \|x_k - x^*\|^2 - \|x_{k+1} - x_k\|^2$$

and the sequence $\{x_k\}$ is bounded. Furthermore, either $\{x_k\}$ is finite and the last iterate is a solution of (1.1) or $\{x_k\}$ is infinite and

$$\sum_{k=0}^{\infty} \|x_{k+1} - x_k\|^2 < \infty,$$

which means

$$\lim_{k\to\infty} \|x_{k+1} - x_k\| = 0. \tag{3.2}$$

Proof. The conclusion follows from Theorem 2.1 in [16]. □

Since $F(x)$ is continuous and $\{x_k\}$ is a bounded sequence it follows that there exists a constant $M > 0$ such that $\|F(x_k)\| \leq M$, $\forall k \geq 0$. Thus, $\{F_k\}$ is bounded.

Lemma 3. *For all $k \geq 0$, we have*

$$\left(1 - \frac{1}{\mu}\right)\|F_k\| \leq \|d_k\| \leq \left(1 + \frac{2}{\mu}\right)\|F_k\|.$$

Proof. From (3.1) and Cauchy-Schwarz inequality, we have

$$\|d_k\| \geq \left(1 - \frac{1}{\mu}\right) \|F_k\|.$$

From (2.1)–(2.2) we have

$$\begin{aligned} \|d_k\| &\leq \|F_k\| + |\beta_k|\|d_{k-1}\| + |\theta_k|\|w_{k-1}\| \\ &\leq \|F_k\| + \frac{\|F_k\|^2}{\mu\|d_{k-1}\|\|F_k\|}\|d_{k-1}\| + \frac{\|F_k\|\|w_{k-1}\|}{\mu\|w_{k-1}\|^2}\|w_{k-1}\| \\ &= \|F_k\| + \frac{2}{\mu}\|F_k\| = \left(1 + \frac{2}{\mu}\right) \|F_k\|. \end{aligned}$$

Therefore,

$$\left(1 - \frac{1}{\mu}\right) \|F_k\| \leq \|d_k\| \leq \left(1 + \frac{2}{\mu}\right) \|F_k\|.$$

□

Lemma 4. *Let Assumption 1 hold and the sequences $\{x_k\}$ and $\{z_k\}$ be generated by Algorithm* 1*. Then, we have*

$$\alpha_k \geq \min\left\{\kappa, \frac{\rho\mu(\mu-1)}{(L+\sigma)(\mu+2)^2}\right\}.$$

Proof. If $\alpha_k \neq \kappa$, then $\alpha_k{}' = \frac{\alpha_k}{\rho}$ does not satisfy (2.3), that is

$$-F(x_k + \alpha_k{}' d_k)^T d_k < \sigma\alpha_k{}' \parallel d_k \parallel^2 .$$

This together with (2.3) and (3.1) imply that

$$\begin{aligned} \left(1 - \frac{1}{\mu}\right) \|F_k\|^2 &\leq -F_k^T d_k \\ &= (F(x_k + \alpha_k^{'} d_k) - F_k)^T d_k - F(x_k + \alpha_k^{'} d_k)^T d_k \\ &\leq L\alpha_k^{'}\|d_k\|^2 + \sigma\alpha_k^{'}\|d_k\|^2 = (L+\sigma)\alpha_k\rho^{-1}\|d_k\|^2 \\ &\leq (L+\sigma)\alpha_k\rho^{-1}\left(\frac{\mu+2}{\mu}\right)^2 \|F_k\|^2. \end{aligned}$$

Thus $\alpha_k \geq \rho\mu(\mu-1)/(L+\sigma)(\mu+2)^2$. □

Theorem 1. *Suppose that Assumption 1 holds, and the sequence $\{x_k\}$ is generated by Algorithm 1. Then we have*

$$\lim_{k\to\infty} \inf \parallel F_k \parallel = 0. \tag{3.3}$$

Proof. We assume that (3.3) does not hold, that is, $\exists\, \eta > 0$ such that $\|F_k\| \geq \eta$, $\forall\, k \geq 0$. From (3.1) and Cauchy-Schwarz inequality, we have

$$\|d_k\| \geq (1 - 1/\mu)\, \|F_k\| \geq (1 - 1/\mu)\, \eta > 0, \qquad \forall\, k \geq 0,$$

and (3.2) implies that

$$\lim_{k\to\infty} \alpha_k = 0. \tag{3.4}$$

On the other hand, Lemma 4 implies that $\alpha_k \geq \min\left\{\kappa, \frac{\rho\mu(\mu-1)}{(L+\sigma)(\mu+2)^2}\right\}$, which contradicts (3.4). $\square$

4 Numerical results

In this section, we do some numerical experiments to test the performance of Algorithm 1, herein denoted as $TCGM$, and compare it with the derivative-free projection method $ITDM$ [1] and $M3TFR2$ method [14]. All the algorithms are coded in MATLAB R2016a. In our experiments, the algorithms are stopped whenever the inequality $\|F_k\| \leq 10^{-5}$ is satisfied, or the total number of iterations exceeds 5000. The parameters used in $ITDM$ and $M3TFR2$ methods are set as in respective papers. The parameters in $TCGM$ are selected as $\sigma = 10^{-4}$, $\rho = 0.5$, $r = 10^{-3}$, $\mu = 1.3$ and $\kappa = 1$. All the algorithms are tested using the following test problems with different initial starting points and various dimensions.

Problem 1. [4]

$$F_i(x) = 2c(x_i - 1) + 4x_i \sum_{i=1}^{n} x_i^2 - x_i, \quad \text{for} \quad i = 1, 2, 3, ..., n, \text{ and } c = 10^{-5}.$$

Problem 2. [11]

$$F(x) = Ax + g(x),$$

where $g(x) = (e^{x_1} - 1, e^{x_2} - 1, ..., e^{x_n} - 1)^T$ and

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

Problem 3. [1]

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

Problem 4. [19]

$$F_i(x) = e^{x_i} - 2, \quad \text{for} \quad i = 1, 2, 3, ..., n.$$

Problem 5. [12]

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

Problem 6. [21]

$$
\begin{aligned}
F_{2i-1}(x) &= x_{2i-1} + ((5 - x_{2i})x_{2i} - 2)x_{2i} - 13,\\
F_{2i}(x) &= x_{2i-1} + ((1 + x_{2i})x_{2i} - 14)x_{2i} - 29,
\end{aligned}
$$

for $i = 1, 2, 3, ..., \frac{n}{2}(n \text{ even})$.

Problem 7. [21]

$$
\begin{aligned}
F_1(x) &= 2x_1 + 0.5h^2(x_1 + h)^3 - x_2,\\
F_i(x) &= 2x_i + 0.5h^2(x_i + ih)^3 - x_{i-1} + x_{i+1}, \text{ for} \quad i = 2, 3, ..., n-1,\\
F_n(x) &= 2x_n + 0.5h^2(x_n + nh)^3 - x_{n-1},
\end{aligned}
$$

where $h = 1/(n+1)$.

Problem 8. [2]

$$
F_i(x) = 2x_i - \sin(|x_i|), \quad \text{for} \quad i = 1, 2, 3, ..., n.
$$

Problem 9. [21]

$$
\begin{aligned}
F_1(x) &= 3x_1^3 + 2x_2 - 5 + \sin(x_1 - x_2)\sin(x_1 + x_2),\\
F_i(x) &= -x_{i-1}e^{x_{i-1} - x_i} + x_i(4 + 3x_i^2) + 2x_{i+1}\\
&\qquad + \sin(x_i - x_{i+1})\sin(x_i + x_{i+1}) - 8, \text{ for} \quad i = 2, 3, ..., n-1,\\
F_n(x) &= -x_{n-1}e^{x_{n-1} - x_n} + 4x_n - 3.
\end{aligned}
$$

Problem 10. [2]

$$
\begin{aligned}
F_1(x) &= 2x_1 - \sin(x_1) - 1,\\
F_i(x) &= -2x_{i-1} + 2x_i + \sin(x_i) - 1, \quad \text{for} \quad i = 1, 2, 3, ..., n-1,\\
F_n(x) &= 2x_n + \sin(x_n) - 1.
\end{aligned}
$$

The results are presented in Tables 1–10, where $x_0^1 = (1, 1, ..., 1)^T$, $x_0^2 = (-1, -1, ..., -1)^T$, $x_0^3 = (0.1, 0.1, ..., 0.1)^T$ and $x_0^4 = (-0.1, -0.1, ..., -0.1)^T$. In each table, we report the dimension of the problem (DIM), the number of iterations (NI), the number of function evaluations (FE) and the CPU time in seconds. We note here that all algorithms managed to solve all the ten test functions successfully. We see that the proposed method performs better than the other methods in almost all the problems.

To have comprehensive comparisons for these methods with respect to number of iterations, number of function evaluations and the CPU time, we apply the performance profiles tool of Dolan and Moré [6] to obtain Figures 1–2. The

Table 1. Numerical results of Problem 1.

		NI			FE			CPU		
x_0	DIM	TCGM	M3TFR2	ITDM	TCGM	M3TFR2	ITDM	TCGM	M3TFR2	ITDM
x_0^1	3000	9	13	13	29	57	33	0.0028	0.0066	0.0045
	5000	9	18	13	29	85	33	0.0041	0.0122	0.0068
	10000	9	22	10	29	125	30	0.0085	0.0340	0.0122
	20000	9	28	10	29	191	30	0.0218	0.1214	0.0216
x_0^2	3000	9	13	13	29	57	33	0.0029	0.0056	0.0045
	5000	9	16	13	29	78	33	0.0040	0.0110	0.0069
	10000	9	21	10	29	122	30	0.0108	0.0425	0.0147
	20000	9	28	10	29	191	30	0.0144	0.1033	0.0249
x_0^3	3000	8	16	13	20	80	34	0.0019	0.0075	0.0044
	5000	8	19	13	20	106	34	0.0029	0.0148	0.0070
	10000	8	22	14	20	149	37	0.0085	0.0446	0.0196
	20000	8	29	14	20	223	37	0.0157	0.1245	0.0351
x_0^4	3000	8	17	13	20	83	34	0.0030	0.0118	0.0048
	5000	8	17	13	20	99	34	0.0029	0.0138	0.0068
	10000	8	23	14	20	152	37	0.0054	0.0395	0.0138
	20000	8	27	14	20	217	37	0.0104	0.1176	0.0266

Table 2. Numerical results of Problem 2.

		NI			FE			CPU		
x_0	DIM	TCGM	M3TFR2	ITDM	TCGM	M3TFR2	ITDM	TCGM	M3TFR2	ITDM
x_0^1	300	17	17	22	45	58	72	0.0085	0.0298	0.0140
	500	17	19	23	44	71	75	0.0296	0.0772	0.0537
	1000	17	21	23	45	93	112	0.0913	0.1997	0.2398
	2000	16	24	23	40	125	75	0.3556	1.1285	0.6854
x_0^2	300	17	21	23	40	80	72	0.0069	0.0146	0.0127
	500	18	21	26	44	89	121	0.0276	0.0561	0.0772
	1000	21	22	25	58	109	80	0.1163	0.2198	0.1708
	2000	19	25	24	48	155	74	0.4311	1.4178	0.6819
x_0^3	300	14	16	18	36	46	55	0.0061	0.0087	0.0097
	500	15	17	20	39	48	64	0.0242	0.0299	0.0411
	1000	14	17	20	36	50	64	0.0707	0.1000	0.1301
	2000	14	17	19	35	48	57	0.3142	0.4364	0.5250
x_0^4	300	14	16	18	37	47	58	0.0071	0.0094	0.0113
	500	14	16	19	36	47	61	0.0249	0.0333	0.0437
	1000	16	17	19	42	48	61	0.0831	0.0970	0.1285
	2000	15	17	19	38	50	61	0.3447	0.4539	0.5635

Table 3. Numerical results of Problem 3.

		NI			FE			CPU		
x_0	DIM	TCGM	M3TFR2	ITDM	TCGM	M3TFR2	ITDM	TCGM	M3TFR2	ITDM
x_0^1	300	8	15	15	16	81	29	0.0006	0.0093	0.0014
	500	8	18	16	16	108	31	0.0009	0.0055	0.0020
	1000	8	22	16	16	158	31	0.0014	0.0138	0.0034
	2000	8	29	16	16	242	31	0.0026	0.0386	0.0061
x_0^2	300	8	29	16	16	207	31	0.0006	0.0075	0.0014
	500	8	36	16	16	285	31	0.0009	0.0145	0.0020
	1000	9	45	17	18	420	33	0.0016	0.0365	0.0036
	2000	9	61	17	18	639	33	0.0029	0.1014	0.0065
x_0^3	300	8	20	16	16	131	31	0.0006	0.0047	0.0014
	500	8	28	16	16	188	31	0.0009	0.0096	0.0020
	1000	8	33	16	16	276	31	0.0014	0.0243	0.0034
	2000	9	46	17	18	423	33	0.0029	0.0673	0.0065
x_0^4	300	8	27	16	16	164	31	0.0006	0.0060	0.0014
	500	8	26	16	16	198	31	0.0009	0.0101	0.0020
	1000	9	35	16	18	300	31	0.0016	0.0261	0.0034
	2000	9	49	17	18	463	33	0.0029	0.0737	0.0065

Table 4. Numerical results of Problem 4.

		NI			FE			CPU		
x_0	DIM	TCGM	M3TFR2	ITDM	TCGM	M3TFR2	ITDM	TCGM	M3TFR2	ITDM
x_0^1	300	7	5	10	20	11	28	0.0005	0.0030	0.0008
	500	7	5	10	20	13	28	0.0005	0.0004	0.0010
	1000	7	6	11	20	18	31	0.0007	0.0007	0.0015
	2000	8	8	11	23	28	31	0.0012	0.0014	0.0022
x_0^2	300	7	19	10	18	104	26	0.0004	0.0024	0.0008
	500	7	21	11	18	135	29	0.0005	0.0034	0.0011
	1000	8	26	11	21	193	29	0.0008	0.0062	0.0014
	2000	8	35	11	21	295	29	0.0010	0.0130	0.0021
x_0^3	300	8	13	10	22	46	27	0.0005	0.0011	0.0008
	500	8	12	11	22	43	30	0.0006	0.0012	0.0011
	1000	8	14	11	22	60	30	0.0008	0.0020	0.0014
	2000	8	18	11	22	87	30	0.0011	0.0040	0.0021
x_0^4	300	8	13	11	22	51	30	0.0005	0.0012	0.0009
	500	8	15	11	22	62	30	0.0006	0.0016	0.0010
	1000	8	16	12	22	78	33	0.0008	0.0026	0.0015
	2000	9	21	12	25	118	33	0.0012	0.0053	0.0023

Table 5. Numerical results of Problem 5.

		NI			FE			CPU		
x_0	DIM	TCGM	M3TFR2	ITDM	TCGM	M3TFR2	ITDM	TCGM	M3TFR2	ITDM
x_0^1	3000	17	27	22	45	159	69	0.0097	0.0326	0.0165
	5000	19	33	23	50	222	72	0.0142	0.0724	0.0258
	10000	18	41	21	46	331	64	0.0205	0.1862	0.0343
	20000	17	55	25	42	507	80	0.0416	0.5543	0.0997
x_0^2	3000	19	30	26	48	200	81	0.0104	0.0447	0.0198
	5000	19	32	25	47	252	78	0.0160	0.0680	0.0301
	10000	19	44	28	47	389	129	0.0279	0.2348	0.0753
	20000	17	57	26	40	577	83	0.0705	0.6020	0.1022
x_0^3	3000	14	17	20	35	51	61	0.0051	0.0080	0.0119
	5000	14	16	20	35	46	61	0.0080	0.0144	0.0213
	10000	14	17	19	34	52	94	0.0153	0.0238	0.0509
	20000	14	13	20	34	44	59	0.0287	0.0423	0.0603
x_0^4	3000	15	18	15	38	53	44	0.0056	0.0078	0.0076
	5000	15	17	21	38	50	70	0.0087	0.0115	0.0188
	10000	16	18	19	41	58	59	0.0225	0.0259	0.0405
	20000	16	17	20	42	58	63	0.0470	0.0496	0.0793

Table 6. Numerical results of Problem 6.

		NI			FE			CPU		
x_0	DIM	TCGM	M3TFR2	ITDM	TCGM	M3TFR2	ITDM	TCGM	M3TFR2	ITDM
x_0^1	300	181	204	624	984	878	9849	0.0193	0.0220	0.1957
	500	125	210	1083	678	945	17573	0.0166	0.0253	0.4370
	1000	155	128	384	835	806	5937	0.0301	0.0305	0.2251
	2000	231	196	881	1296	1285	14023	0.0770	0.0812	0.8993
x_0^2	300	185	197	794	1030	895	12665	0.0205	0.0189	0.2527
	500	97	194	930	515	1014	14747	0.0126	0.0263	0.3688
	1000	189	218	729	1046	1365	11549	0.0374	0.0516	0.4370
	2000	124	270	600	675	1874	9327	0.0400	0.1167	0.5942
x_0^3	300	198	252	524	1093	1075	8202	0.0217	0.0228	0.1625
	500	144	158	832	772	852	13048	0.0189	0.0221	0.3267
	1000	154	242	845	842	1314	13223	0.0302	0.0504	0.5015
	2000	193	229	680	1068	1543	10564	0.0628	0.0958	0.6666
x_0^4	300	117	157	824	622	775	13077	0.0123	0.0162	0.2595
	500	114	303	825	604	1350	13067	0.0147	0.0355	0.3267
	1000	86	235	815	454	1283	12914	0.0163	0.0495	0.4895
	2000	109	231	868	578	1606	13877	0.0343	0.0998	0.8735

Table 7. Numerical results of Problem 7.

		NI			FE			CPU		
x_0	DIM	TCGM	M3TFR2	ITDM	TCGM	M3TFR2	ITDM	TCGM	M3TFR2	ITDM
x_0^1	300	60	119	37	178	370	140	0.0117	0.0246	0.0096
	500	60	112	37	178	357	139	0.0178	0.0360	0.0145
	1000	60	108	36	178	364	134	0.0330	0.0680	0.0259
	2000	60	100	33	178	369	122	0.0632	0.1323	0.0459
x_0^2	300	60	119	37	178	370	140	0.0119	0.0250	0.0097
	500	60	112	37	178	357	139	0.0178	0.0361	0.0145
	1000	60	108	36	178	364	134	0.0329	0.0679	0.0258
	2000	60	100	33	178	369	122	0.0631	0.1319	0.0453
x_0^3	300	43	76	29	127	227	106	0.0085	0.0152	0.0074
	500	43	74	28	127	221	105	0.0127	0.0223	0.0109
	1000	43	72	29	127	215	106	0.0234	0.0401	0.0205
	2000	43	70	28	127	209	103	0.0450	0.0748	0.0381
x_0^4	300	43	76	29	127	227	106	0.0084	0.0151	0.0073
	500	43	74	28	127	221	105	0.0128	0.0224	0.0110
	1000	43	72	29	127	215	106	0.0234	0.0402	0.0204
	2000	43	70	28	127	209	103	0.0450	0.0748	0.0382

Table 8. Numerical results of Problem 8.

		NI			FE			CPU		
x_0	DIM	TCGM	M3TFR2	ITDM	TCGM	M3TFR2	ITDM	TCGM	M3TFR2	ITDM
x_0^1	3000	8	23	16	16	160	31	0.0011	0.0092	0.0034
	5000	8	27	16	16	209	31	0.0015	0.0164	0.0050
	10000	8	39	16	16	337	31	0.0028	0.0434	0.0092
	20000	9	52	17	18	511	33	0.0053	0.1367	0.0179
x_0^2	3000	15	26	15	57	180	57	0.0040	0.0129	0.0056
	5000	15	31	16	57	234	61	0.0054	0.0229	0.0084
	10000	15	43	16	57	362	61	0.0080	0.0532	0.0129
	20000	15	58	16	57	553	61	0.0136	0.1349	0.0255
x_0^3	3000	7	4	13	14	10	25	0.0012	0.0008	0.0034
	5000	7	5	14	14	14	27	0.0015	0.0014	0.0048
	10000	8	7	14	16	24	27	0.0028	0.0037	0.0083
	20000	8	7	14	16	28	27	0.0048	0.0071	0.0147
x_0^4	3000	12	10	13	45	29	49	0.0030	0.0022	0.0046
	5000	12	9	13	45	27	49	0.0037	0.0025	0.0062
	10000	13	10	14	49	34	53	0.0060	0.0047	0.0106
	20000	13	12	14	49	44	53	0.0113	0.0109	0.0203

Table 9. Numerical results of Problem 9.

		NI			FE			CPU		
x_0	DIM	TCGM	M3TFR2	ITDM	TCGM	M3TFR2	ITDM	TCGM	M3TFR2	ITDM
x_0^1	3000	1	1	1	1	1	1	0.0002	0.0002	0.0002
	5000	1	1	1	1	1	1	0.0003	0.0004	0.0003
	10000	1	1	1	1	1	1	0.0007	0.0007	0.0007
	20000	1	1	1	1	1	1	0.0013	0.0013	0.0013
x_0^2	3000	19	53	19	93	447	111	0.0221	0.1023	0.0289
	5000	19	65	23	93	586	136	0.0529	0.2467	0.0640
	10000	19	81	23	93	839	138	0.0852	0.6982	0.1223
	20000	19	109	24	93	1266	147	0.1505	2.1338	0.3055
x_0^3	3000	21	31	19	106	187	110	0.0431	0.0489	0.0282
	5000	21	35	19	106	243	110	0.0475	0.1227	0.0466
	10000	21	46	21	105	368	124	0.1213	0.3983	0.1271
	20000	21	60	19	107	544	111	0.2026	0.9904	0.2261
x_0^4	3000	19	34	22	91	230	131	0.0412	0.0693	0.0356
	5000	21	39	20	103	299	117	0.0614	0.1698	0.0552
	10000	20	50	20	98	439	119	0.0856	0.3457	0.1516
	20000	20	68	21	98	667	124	0.1671	1.2313	0.2964

Table 10. Numerical results of Problem 10.

		NI			FE			CPU		
x_0	DIM	TCGM	M3TFR2	ITDM	TCGM	M3TFR2	ITDM	TCGM	M3TFR2	ITDM
x_0^1	3000	47	55	46	156	231	240	0.0182	0.0293	0.0438
	5000	47	60	49	156	282	270	0.0390	0.0554	0.0783
	10000	47	68	47	158	371	203	0.0638	0.1485	0.1183
	20000	47	80	53	158	515	287	0.1047	0.4317	0.2471
x_0^2	3000	50	66	56	163	324	266	0.0226	0.0383	0.0470
	5000	50	75	56	165	412	312	0.0382	0.0929	0.1007
	10000	50	86	53	165	558	228	0.0632	0.2602	0.1246
	20000	51	105	54	170	799	282	0.1357	0.6699	0.3113
x_0^3	3000	44	43	54	142	129	278	0.0179	0.0255	0.0422
	5000	44	43	52	142	130	255	0.0252	0.0241	0.0527
	10000	44	44	45	142	137	274	0.0804	0.0732	0.1359
	20000	44	47	50	144	149	244	0.0962	0.1259	0.2088
x_0^4	3000	46	46	47	150	150	210	0.0237	0.0181	0.0287
	5000	46	49	47	150	168	220	0.0347	0.0328	0.0489
	10000	47	52	50	155	192	315	0.0682	0.0813	0.1844
	20000	46	56	50	152	233	228	0.1068	0.1990	0.2056

Dolan and Moré [6] performance profiles procedure is a tool for evaluating and comparing the performances of iterative methods. The profile of each method is measured according to the ratio of its computational outcome compared to the computational outcome of the best presented method. Figures 1–2 clearly show that *TCGM* method is the most efficient as compared to the other two methods.

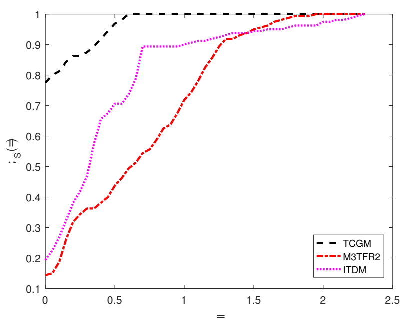
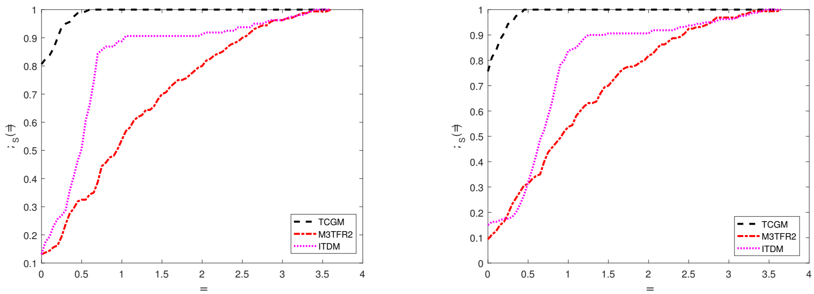


Figure 1. Iterations performance profile

5 Conclusions

In this paper, we proposed a three-term conjugate gradient-based method (*TCGM*) for solving systems of large-scale nonlinear monotone equations. The proposed method is free from derivative evaluations and also satisfies the sufficient descent condition independent of any line search. Its global convergence was also established. The proposed algorithm was tested on some benchmark problems with different starting points and different dimensions and the numerical results show that the method is very efficient.



a) b)

Figure 2. Results of numerical experiments: a) function evaluations performance profile b) CPU time performance profile

References

[1] A.B. Abubakar and P. Kumam. An improved three-term derivative-free method for solving nonlinear equations. *Comp. Appl. Math.*, **37**(5):6760–6773, 2018. https://doi.org/10.1007/s40314-018-0712-5.

[2] A.B. Abubakar and P. Kumam. A descent Dai-Liao conjugate gradient method for nonlinear equations. *Numer. Algor.*, **81**(1):197–210, 2019. https://doi.org/10.1007/s11075-018-0541-z.

[3] B. Baluch, Z. Salleh, A. Alhawarat and U.A.M. Roslan. A new modified three-term conjugate gradient method with sufficient descent property and its global convergence. *J. Math.*, **2017**(Article ID 2715854):12 pages, 2017. https://doi.org/10.1155/2017/2715854.

[4] Y. Ding, Y. Xiao and J. Li. A class of conjugate gradient methods for convex constrained monotone equations. *Optim.*, **66**(12):2309–2328, 2017. https://doi.org/10.1080/02331934.2017.1372438.

[5] S.P. Dirkse and M.C. Ferris. MCPLIB: A collection of nonlinear mixed complimentary problems. *Optim. Methods Softw.*, **5**:319–345, 1995. https://doi.org/10.1080/10556789508805619.

[6] E.D. Dolan and J.J. More. Benchmarking optimization software with performance profiles. *Math. Program.*, **91**(2):201–213, 2002. https://doi.org/10.1007/s101070100263.

[7] D. Feng, M. Sun and X. Wang. A family of conjugate gradient methods for large-scale nonlinear equations. *J. Inequal. Appl.*, **2017**(236):1–8, 2017. https://doi.org/10.1186/s13660-017-1510-0.

[8] Q. Hu, Y. Xiao and Q. Wang. Non-smooth equations based methods for l_1-norm problems with applications to compressed sensing. *Nonlinear Anal.*, **74**:3570–3577, 2011. https://doi.org/10.1016/j.na.2011.02.040.

[9] M.A.H. Ibrahim, M. Mamat and W.J. Leong. The hybrid BFGS-CG method in solving unconstrained optimization problems. *Abs. Appl. Anal.*, **2014**(Article ID 507102):6 pages, 2014. https://doi.org/10.1155/2014/507102.

[10] J. Liu and Y. Feng. A derivative-free iterative method for nonlinear monotone equations with convex constraints. *Numer. Algor.*, **82**(1):245–262, 2019. https://doi.org/10.1007/s11075-018-0603-2.

[11] J. Liu and S. Li. Spectral DY-type projection method for nonlinear monotone system of equations. *J. Comput. Math.*, **33**(4):341–355, 2015. https://doi.org/10.4208/jcm.1412-m4494.

[12] J. Liu and S. Li. A three-term derivative-free projection method for nonlinear monotone system of equations. *Calcolo*, **53**(3):427–450, 2016. https://doi.org/10.1007/s10092-015-0156-x.

[13] K. Meintjes and A.P. Morgan. A methodology for solving chemical equilibrium systems. *Appl. Math. Comput.*, **22**:333–361, 1987. https://doi.org/10.1016/0096-3003(87)90076-2.

[14] Z. Papp and S. Rapajić. FR type methods for systems of large-scale nonlinear monotone equations. *Appl. Math. Comput.*, **269**:816–823, 2015. https://doi.org/10.1016/j.amc.2015.08.002.

[15] M.A.K. Shiker and K. Amini. A new projection-based algorithm for solving a large-scale nonlinear system of monotone equations. *Croat. Oper. Res. Rev*, **9**:63–73, 2018. https://doi.org/10.17535/crorr.2018.0006.

[16] 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*, volume 22 of *Applied Optimization*, pp. 355–369, Boston, MA, 1998. Springer. https://doi.org/10.1007/978-1-4757-6388-1_18.

[17] P.S. Stanimirović, B. Ivanov, S. Djordjević and I. Brajević. New hybrid conjugate gradient and Broyden-Fletcher-Goldfarb-Shanno conjugate gradient methods. *J. Optim. Theory Appl.*, **178**(3):860–884, 2018. https://doi.org/10.1007/s10957-018-1324-3.

[18] M. Sun and J. Liu. Three derivative-free projection methods for nonlinear equations with convex constraints. *J. Appl. Math. Comp.*, **47**(1-2):265–276, 2015. https://doi.org/10.1007/s12190-014-0774-5.

[19] S. Wang and H. Guan. A scaled conjugate gradient method for solving monotone nonlinear equations with convex constraints. *J. Appl. Math.*, **2013**(Article ID 286486):7 pages, 2013. https://doi.org/10.1155/2013/286486.

[20] X.Y. Wang, S.J. Li and X.P. Kou. A self-adaptive three-term conjugate gradient method for monotone nonlinear equations with convex constraints. *Calcolo*, **53**(2):133–145, 2016. https://doi.org/10.1007/s10092-015-0140-5.

[21] G. Yuan and M. Zhang. A three-terms Polak-Ribiere-Polyak conjugate gradient algorithm for large-scale nonlinear equations. *J. Comput. Appl. Math.*, **286**:186–195, 2015. https://doi.org/10.1016/j.cam.2015.03.014.

[22] Y. Zheng and B. Zheng. Two new Dai-Liao-type conjugate gradient methods for unconstrained optimization problems. *J. Optim. Theory. Appl.*, **175**(2):502–509, 2017. https://doi.org/10.1007/s10957-017-1140-1.