Journal of the Egyptian Mathematical Society

ORIGINAL RESEARCH Open Access

Self adaptive spectral conjugate gradient method for solving nonlinear monotone equations

M. Koorapetse and P. Kaelo*

*Correspondence:
kaelop@mopipi.ub.bw
Department of Mathematics, University of Botswana, Private Bag UB00704 Gaborone, Botswana

Abstract

In this paper, we propose a self adaptive spectral conjugate gradient-based projection method for systems of nonlinear monotone equations. Based on its modest memory requirement and its efficiency, the method is suitable for solving large-scale equations. We show that the method satisfies the descent condition $F_k^T d_k \leq -c\|F_k\|^2, c > 0$, and also prove its global convergence. The method is compared to other existing methods on a set of benchmark test problems and results show that the method is very efficient and therefore promising.

Keywords: Self adaptive, Spectral conjugate gradient method, Nonlinear monotone equations

AMS Subject Classification: 90C06, 90C30, 90C56, 65K05, 65K10

Introduction

In this paper, we focus on solving large-scale nonlinear system of equations

$$F(x) = 0, \tag{1}$$

where $F : \mathbb{R}^n \to \mathbb{R}^n$ is continuous and monotone. A function F is monotone if it satisfies the monotonicity condition

$$(F(x) - F(y))^T (x - y) \geq 0, \quad \forall x, y \in \mathbb{R}^n. \tag{2}$$

Nonlinear monotone equations arise in many practical applications, for example, chemical equilibrium systems [1], economic equilibrium problems [2], and some monotone variational inequality problems [3]. A number of computational methods have been proposed to solve nonlinear equations. Among them, Newton's method, quasi-Newton method, Gauss-Newton method, and their variants are very popular due to their local superlinear convergence property (see, for example, [4–9]). However, they are not suitable for large-scale nonlinear monotone equations as they need to solve a linear system of equations using the second derivative information (Jacobian matrix or an approximation of it).

Due to their modest memory requirements, conjugate gradient-based projection methods are suitable for solving large-scale nonlinear monotone equations (1). Conjugate

gradient-based projection methods generate a sequence $\{x_k\}$ by exploring the monotonicity of the function F. Let $z_k = x_k + \alpha_k d_k$, where $\alpha_k > 0$ is the step length that is determined by some line search and

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

$F_k = F(x_k)$ and β_k is a parameter, is the search direction. Then by monotonicity of F, the hyperplane

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

strictly separates the current iterate x_k from the solution set of (1). Projecting x_k on this hyperplane generates the next iterate x_{k+1} as

$$x_{k+1} = x_k - \frac{F(z_k)^T (x_k - z_k)}{\|F(z_k)\|^2} F(z_k). \tag{3}$$

This projection concept on the hyperplane H_k was first presented by Solodov and Svaiter [10].

Following Solodov and Svaiter [10], a lot of work has been done, and continues to be done, to come up with a number of conjugate gradient-based projection methods for nonlinear monotone equations. For example, Hu and Wei [11] proposed a conjugate gradient-based projection method for nonlinear monotone equations (1) where the search direction d_k is given as

$$d_k = \begin{cases} -F_k, & k = 0, \\ -F_k + \frac{F_k^T y_{k-1} d_{k-1} - d_{k-1}^T F_k y_{k-1}}{\max\left(\gamma \|d_{k-1}\| \|y_{k-1}\|, d_{k-1}^T y_{k-1}, -d_{k-1}^T F_{k-1}\right)}, & k \geq 1, \end{cases}$$

$y_{k-1} = F_k - F_{k-1}$ and $\gamma > 0$. This method was shown to perform well numerically and its global convergence was established using the line search

$$-F(z_k)^T d_k \geq \sigma \alpha_k \|F(z_k)\| \|d_k\|^2, \tag{4}$$

with $\sigma > 0$ being a constant.

Recently, three term conjugate gradient-based projection methods have also been presented. One such method is that by Feng et al. [12] who presented their direction as

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

where $|\beta_k| \leq t \frac{\|F_k\|}{\|d_{k-1}\|}$, $\forall k \geq 1$, and $t > 0$ is a constant. The global convergence of this method was also established using the line search (4). For other conjugate gradient-based projection methods, the reader is referred to [13–27].

In this paper, following the work of Abubakar and Kumam [21], Hu and Wei [11] and that of Liu and Li [22], we propose a self adaptive spectral conjugate gradient-based projection method for solving systems of nonlinear monotone Eq. (1). This method is presented in the next section and the rest of the paper is organized as follows. In "Convergence analysis" section, we show that the proposed method satisfies the descent property $F_k^T d_k \leq -c\|F_k\|^2, c > 0$, and also establish its global convergence. In "Numerical experiments" section, we present the numerical results and lastly, conclusion is presented in "Conclusion" section.

Algorithm

In this section, we give the details of the proposed method. We start by briefly reviewing the work of Abubakar and Kumam [21] and that of Liu and Li [22].

Most recently, Abubakar and Kumam [21] proposed the direction

$$d_k = \begin{cases} -F_k, & k = 0, \\ -F_k + \frac{F_k^T w_{k-1} d_{k-1} - F_k^T d_{k-1} w_{k-1}}{\max\left(\mu \|d_{k-1}\| \|w_{k-1}\|, w_k^T d_{k-1}\right)}, & k \geq 1, \end{cases}$$

where μ is a positive constant and

$$w_{k-1} = y_{k-1} + t\|F_k\| s_{k-1}, \quad t = 1 + \|F_k\|^{-1} \max\left(0, -\frac{y_{k-1}^T s_{k-1}}{\|s_{k-1}\|^2}\right)$$

and $s_{k-1} = z_{k-1} - x_{k-1} = \alpha_{k-1} d_{k-1}$. This method was shown to perform well numerically and its global convergence was established using line search (4). In 2015, Liu and Li [22] proposed a spectral DY-type projection method for nonlinear monotone system of Eq. (1) with the search direction d_k as

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

where $\beta_k^{DY} = \frac{\|F_k\|^2}{d_{k-1}^T u_{k-1}}$, $u_{k-1} = y_{k-1} + t d_{k-1}$, $t = 1 + \max\left\{0, -\frac{d_{k-1}^T y_{k-1}}{d_{k-1}^T d_{k-1}}\right\}$, $y_{k-1} = F_k - F_{k-1} + rs_{k-1}$ with $s_{k-1} = x_k - x_{k-1}$, $r > 0$ being a constant and $\lambda_k = \frac{s_{k-1}^T s_{k-1}}{s_{k-1}^T y_{k-1}}$. The global convergence of this method was established using the line search

$$-F(z_k)^T d_k \geq \sigma \alpha_k \|d_k\|^2. \tag{5}$$

Motivated by the work of Abubakar and Kumam [21], Hu and Wei [11] and that of Liu and Li [22], in this paper we present our direction as

$$d_k = \begin{cases} -F_k, & k = 0, \\ -\lambda_k^* F_k + \beta_k^{MP} d_{k-1} - \delta_k^{MP} y_{k-1}, & k \geq 1, \end{cases} \tag{6}$$

where

$$\beta_k^{MP} = \frac{F_k^T y_{k-1}}{\max\{\mu_k d_{k-1}^T y_{k-1}, -\eta F_{k-1}^T d_{k-1} + \mu_k \|d_{k-1}\| \|y_{k-1}\|\}} \tag{7}$$

and

$$\delta_k^{MP} = \frac{F_k^T d_{k-1}}{\max\{\mu_k d_{k-1}^T y_{k-1}, -\eta F_{k-1}^T d_{k-1} + \mu_k \|d_{k-1}\| \|y_{k-1}\|\}} \tag{8}$$

with $\eta > 0$ being a constant and the parameters $\lambda_k^* = \frac{s_{k-1}^T y_{k-1}}{s_{k-1}^T s_{k-1}}$ and $\mu_k > \frac{1}{\lambda_k^*}$ where $s_{k-1} = x_k - x_{k-1}$ and $y_{k-1} = F_k - F_{k-1} + rs_{k-1}$, $r \in (0, 1)$. With d_k defined by (6), (7), and (8), we now present our algorithm.

Algorithm 1 Self adaptive spectral conjugate gradient-based projection method (SASCGM)

1: Give x_0, the parameters $\sigma \in (0,1)$, $\kappa, r \in (0,1)$, $\epsilon > 0$ and $\rho \in (0,1)$. Set $k = 0$.
2: **for** $k = 0, 1, \ldots$ **do**
3: If $\|F_k\| \leq \epsilon$, then stop. Otherwise, go to Step 4.
4: Compute d_k by (6), (7) and (8).
5: Compute $z_k = x_k + \alpha_k d_k$ where $\alpha_k = \max\{\kappa\rho^i : i = 0, 1, 2, \ldots\}$ such that (5) is satisfied.
6: If $\|F(z_k)\| \leq \|F(x_k)\|$, then set $x_{k+1} = z_k$. Otherwise, compute x_{k+1} using (3).
7: Set $k = k + 1$ and go to Step 3.
8: **end for**

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

Assumption 1
(i) The function $F(\cdot)$ is monotone on $\mathbb{R}^n$, i.e. $(F(x) - F(y))^T(x - y) \geq 0$, $\forall x, y \in \mathbb{R}^n$.
(ii) The solution set of (1) is nonempty.
(iii) The function $F(\cdot)$ is Lipschitz continuous on $\mathbb{R}^n$, i.e. there exists a positive constant L such that

$$\| F(x) - F(y) \| \leq L \| x - y \|, \quad \forall x, y \in \mathbb{R}^n. \tag{9}$$

Convergence analysis

In this section we present the descent property and global convergence of the proposed method.

Lemma 1 *For all $k \geq 0$, we have*

$$r \leq \lambda_k^* \leq L + r. \tag{10}$$

Proof From the definition of y_{k-1}, we get that

$$s_{k-1}^T y_{k-1} = (F_k - F_{k-1})^T (x_k - x_{k-1}) + r\|s_{k-1}\|^2,$$

which using the monotonicity of F it follows that

$$s_{k-1}^T y_{k-1} \geq r\|s_{k-1}\|^2. \tag{11}$$

Also, from the Lipschitz continuity we obtain that

$$s_{k-1}^T y_{k-1} \leq (L + r)\|s_{k-1}\|^2. \tag{12}$$

Combining (11) and (12) we get the inequality (10). This, therefore, means that λ_k^* is well defined. □

Lemma 2 *Suppose that Assumption 1 holds. Let the sequence $\{x_k\}$ be generated by Algorithm 1. Then the search direction d_k satisfies the descent condition*

$$F_k^T d_k \leq -r\|F_k\|^2, \quad \forall k \geq 0. \tag{13}$$

Proof Since $d_0 = -F_0$, we have $F_0^T d_0 = -\|F_0\|^2$, which satisfies (13). For $k \geq 1$, we have from (6) that

$$F_k^T d_k = -\lambda_k^* \|F_k\|^2 + \beta_k^{MP} F_k^T d_{k-1} - \delta_k^{MP} F_k^T y_{k-1}. \tag{14}$$

Using (7) and (8) we obtain

$$\begin{aligned} F_k^T d_k &= -\lambda_k^* \|F_k\|^2 + \frac{\left(F_k^T y_{k-1}\right)\left(F_k^T d_{k-1}\right)}{\max\left\{\mu_k d_{k-1}^T y_{k-1}, -\eta F_{k-1}^T d_{k-1} + \mu_k \|d_{k-1}\| \|y_{k-1}\|\right\}} \\ &\quad - \frac{\left(F_k^T d_{k-1}\right)\left(F_k^T y_{k-1}\right)}{\max\left\{\mu_k d_{k-1}^T y_{k-1}, -\eta F_{k-1}^T d_{k-1} + \mu_k \|d_{k-1}\| \|y_{k-1}\|\right\}} \\ &= -\lambda_k^* \|F_k\|^2 \\ &\leq -r\|F_k\|^2. \end{aligned}$$

□

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

$$r\|F_k\| \leq \|d_k\| \leq 3(L+r)\|F_k\|. \tag{15}$$

Proof From (13) and Cauchy-Schwarz inequality, we have

$$\|d_k\| \geq r\|F_k\|. \tag{16}$$

Also, we have that

$$\max\left\{\mu_k d_{k-1}^T y_{k-1}, -\eta F_{k-1}^T d_{k-1} + \mu_k \|d_{k-1}\| \|y_{k-1}\|\right\} \geq -\eta F_{k-1}^T d_{k-1} + \mu_k \|d_{k-1}\| \|y_{k-1}\|.$$

It then follows from (6), (7), and (8) that

$$\begin{aligned} \|d_k\| &\leq \lambda_k^* \|F_k\| + |\beta_k| \|d_{k-1}\| + |\delta_k| \|y_{k-1}\| \\ &\leq \lambda_k^* \|F_k\| + \frac{\|F_k\| \|y_{k-1}\|}{\mu_k \|d_{k-1}\| \|y_{k-1}\|} \|d_{k-1}\| + \frac{\|F_k\| \|d_{k-1}\|}{\mu_k \|d_{k-1}\| \|y_{k-1}\|} \|y_{k-1}\| \\ &= \lambda_k^* \|F_k\| + \frac{2}{\mu_k} \|F_k\| \\ &\leq 3\lambda_k^* \|F_k\| \\ &\leq 3(L+r)\|F_k\|. \end{aligned}$$

□

Lemma 4 *Suppose Assumption 1 holds and let $\{x_k\}$ be generated by Algorithm 1. Then the steplength α_k is well defined and satisfies the inequality*

$$\alpha_k \geq \min\left\{\kappa, \frac{\rho r}{9(L+\sigma)(L+r)^2}\right\}. \tag{17}$$

Proof Suppose that, at kth iteration, x_k is not a solution, that is, $F_k \neq 0$, and for all $i = 0, 1, 2, \ldots$, inequality (5) fails to hold, that is

$$-F(x_k + \kappa\rho^i d_k)^T d_k < \sigma\kappa\rho^i \parallel d_k \parallel^2. \tag{18}$$

Since F is continuous, taking limits as $i \to \infty$ on both sides of (18) yields

$$-F(x_k)^T d_k \leq 0, \tag{19}$$

which contradicts *Lemma* 2. So, the steplength α_k is well defined and can be determined within a finite number of trials. Now, we prove inequality (17). If $\alpha_k \neq \kappa$, then $\alpha'_k = \frac{\alpha_k}{\rho}$ does not satisfy (5), that is

$$-F(x_k + \alpha'_k d_k)^T d_k < \sigma \alpha'_k \parallel d_k \parallel^2 .$$

Using (9), (13) and (15) we have that

$$\begin{aligned} r\|F_k\|^2 &\leq -F_k^T d_k \\ &= \left(F\left(x_k + \alpha'_k d_k\right) - F_k\right)^T d_k - F\left(x_k + \alpha'_k d_k\right)^T d_k \\ &\leq L\alpha'_k \|d_k\|^2 + \sigma \alpha'_k \|d_k\|^2 \\ &= (L+\sigma)\alpha_k \rho^{-1} \|d_k\|^2 \\ &\leq (L+\sigma)\alpha_k \rho^{-1} \left(3(L+r)\|F_k\|\right)^2 . \end{aligned}$$

Thus

$$\alpha_k \geq \min\left\{\kappa, \frac{\rho r}{9(L+\sigma)(L+r)^2}\right\}.$$

□

The following lemma shows that if the sequence $\{x_k\}$ is generated by *Algorithm* 1, and x^* is a solution of (1), i.e. $F(x^*) = 0$, then the sequence $\{\|x_k - x^*\|\}$ is decreasing and convergent. Thus, the sequence $\{x_k\}$ is bounded.

Lemma 5 *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*

$$\|x_{k+1} - x^*\|^2 \leq \|x_k - x^*\|^2 - \|x_{k+1} - x_k\|^2 \tag{20}$$

and the sequence$\{x_k\}$ *is bounded. Furthermore, either* $\{x_k\}$ *is finite and the last iterate is a solution of* (1), *or* $\{x_k\}$ *is infinite and*

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

which means

$$\lim_{k\to\infty} \|x_{k+1} - x_k\| = 0. \tag{21}$$

Proof The conclusion follows from Theorem 2.1 in [10]. □

Theorem 1 *Let* $\{x_k\}$ *be the sequence generated by Algorithm* 1. *Then*

$$\lim_{k\to\infty} \inf \|F_k\| = 0. \tag{22}$$

Proof Suppose that the inequality (22) is not true. Then there exists a constant $\epsilon_1 > 0$ such that

$$\|F_k\| \geq \epsilon_1, \quad \forall k \geq 0.$$

This together with (13) implies that

$$\|d_k\| \geq r\|F_k\| \geq r\epsilon_1 > 0, \qquad \forall k \geq 0. \tag{23}$$

This and (21) gives that

$$\lim_{k\to\infty} \alpha_k = 0. \tag{24}$$

On the other hand, Lemma 5 implies that

$$\alpha_k \geq \min\left\{\kappa, \frac{\rho r}{9(L+\sigma)(L+r)^2}\right\} > 0,$$

which contradicts (24). Therefore (22) is true. □

Numerical experiments

In this section, results of our proposed method SASCGM are presented together with those of improved three-term derivative-free method (ITDM) [21], the modified Liu-Storey conjugate gradient projection (MLS) method [11], and the spectral DY-type projection method (SDYP) [22]. All algorithms are coded in MATLAB R2016a. In our experiments, we set $\epsilon = 10^{-4}$, i.e., the algorithms are stopped whenever the inequality $\|F_k\| \leq 10^{-4}$ is satisfied, or the total number of iterations exceeds 1000. The method *SASCGM* is implemented with the parameters $\sigma = 10^{-4}$, $\rho = 0.5$, $r = 10^{-3}$, $\mu_k = \frac{1}{\lambda_k^*} + 0.1$ and $\kappa = 1$, while parameters for algorithms ITDM, MLS, and SDYP are set as in respective papers.

The methods are compared using number of iterations, number of function evaluations and CPU time taken for each method to reach the optimal value or termination. We test the algorithms on ten (10) test problems with their dimensions varied from 5000 to 20000, and with four (4) different starting points $x_0 = \left(\frac{1}{n}, \frac{1}{n}, \ldots, \frac{1}{n}\right)^T$, $x_1 = (-1, -1, \ldots, -1)^T$, $x_2 = (0.5, 0.5, \ldots, 0.5)^T$ and $x_3 = (-0.5, -0.5, \ldots, -0.5)^T$. The test functions are listed as follows:

Problem 1. Sun and Liu [19] The mapping F is given by

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

where

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

and $g(x) = (2e^{x_1} - 1, 3e^{x_2} - 1, \ldots, 3e^{x_{n-1}} - 1, 2e^{x_n} - 1)^T$.

Problem 2. Liu and Li [22] Let F be defined by

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

Table 1 Numerical results of Problem 1

SP	DIM	NI				NFE				CPU			
		SASCGM	DLPM	MLS	SDYP	SASCGM	DLPM	MLS	SDYP	SASCGM	DLPM	MLS	SDYP
x_0	5000	19	21	20	18	60	69	62	39	3.9873	4.5649	4.2163	2.5931
	10000	17	20	22	15	50	65	69	30	13.6563	17.7531	18.4265	8.2387
	20000	18	27	21	17	54	98	66	34	61.1241	111.0820	72.5027	38.6235
x_1	5000	18	20	18	16	55	67	58	34	3.8213	4.6693	4.1661	2.3705
	10000	18	19	18	21	52	63	58	56	14.7603	17.7104	16.7535	15.7107
	20000	17	22	21	18	49	74	68	41	56.5528	85.1238	78.3744	47.1014
x_2	5000	20	20	17	16	62	68	51	34	4.6887	4.7276	3.7336	2.3715
	10000	18	20	17	21	58	65	52	49	16.3762	18.3691	14.6588	13.7502
	20000	20	19	21	18	66	59	68	38	75.9882	67.8322	78.6551	44.4579
x_3	5000	18	22	20	19	58	73	63	43	4.0671	5.0897	4.9479	3.0129
	10000	20	23	21	18	61	79	67	40	17.1879	22.1984	18.8722	11.2565
	20000	17	23	23	18	52	77	74	38	60.0939	88.9299	85.4446	43.6871

Table 2 Numerical results of Problem 2

SP	DIM	NI				NFE				CPU			
		SASCGM	DLPM	MLS	SDYP	SASCGM	DLPM	MLS	SDYP	SASCGM	DLPM	MLS	SDYP
x_0	5000	8	7	6	6	17	19	14	13	1.1809	1.3228	1.1286	0.9069
	10000	7	7	5	6	15	19	11	13	4.2963	5.3774	3.1084	3.6528
	20000	7	6	5	6	15	15	11	13	17.1297	17.3641	12.7289	14.7579
x_1	5000	19	24	20	22	57	78	60	54	3.9909	5.4833	4.3748	3.7601
	10000	19	29	22	26	52	119	70	68	14.6032	33.9007	19.6976	19.0850
	20000	21	25	21	25	57	83	63	64	65.7284	95.5594	72.3957	73.8244
x_2	5000	18	22	21	18	52	75	66	43	3.6378	5.2738	5.0433	3.0073
	10000	18	22	17	21	58	72	50	55	16.3369	20.2522	14.0884	15.5107
	20000	18	25	17	17	57	104	51	35	65.4977	119.4127	58.7641	40.4790
x_3	5000	19	22	21	18	56	72	66	38	4.1432	5.0871	4.9204	2.7117
	10000	19	24	19	20	56	80	57	44	15.7323	22.3916	15.9898	12.3388
	20000	19	25	21	17	55	83	65	35	63.0373	94.0844	74.6803	39.6888

Table 3 Numerical results of Problem 3

SP	DIM	NI				NFE				CPU			
		SASCGM	DLPM	MLS	SDYP	SASCGM	DLPM	MLS	SDYP	SASCGM	DLPM	MLS	SDYP
x_0	5000	5	21	20	*	17	129	122	*	0.0951	0.0555	0.1255	*
	10000	5	19	19	*	17	117	116	*	0.0219	0.1717	0.1598	*
	20000	5	19	19	*	17	117	116	*	0.0391	0.3119	0.1863	*
x_1	5000	5	22	18	*	16	136	109	*	0.0068	0.0586	0.0464	*
	10000	5	22	22	*	16	138	136	*	0.0137	0.1423	0.1151	*
	20000	5	23	21	*	16	147	131	*	0.0260	0.2613	0.2114	*
x_2	5000	14	19	18	*	109	119	110	*	0.0592	0.0635	0.0469	*
	10000	14	17	18	*	105	106	111	*	0.0844	0.0922	0.0938	*
	20000	14	19	17	*	106	118	104	*	0.2302	0.2238	0.2718	*
x_3	5000	4	20	18	*	9	124	109	*	0.0040	0.0530	0.0465	*
	10000	4	20	18	*	9	122	109	*	0.0080	0.1049	0.0926	*
	20000	4	22	20	*	9	138	123	*	0.0150	0.2312	0.2057	*

Table 4 Numerical results of Problem 4

SP	DIM	NI				NFE				CPU			
		SASCGM	DLPM	MLS	SDYP	SASCGM	DLPM	MLS	SDYP	SASCGM	DLPM	MLS	SDYP
x_0	5000	8	7	6	6	17	19	14	13	0.0047	0.0054	0.0098	0.0031
	10000	7	7	5	6	15	19	11	13	0.0082	0.0104	0.0059	0.0061
	20000	7	6	5	6	15	15	11	13	0.0150	0.0155	0.0109	0.0111
x_1	5000	19	24	20	22	57	78	60	54	0.0144	0.0215	0.0160	0.0124
	10000	19	29	22	26	52	119	70	68	0.0269	0.0617	0.0356	0.0311
	20000	21	25	21	25	57	83	63	64	0.0553	0.0836	0.0611	0.0549
x_2	5000	18	22	21	18	52	75	66	43	0.0133	0.0204	0.0173	0.0099
	10000	18	22	17	21	58	72	50	55	0.0289	0.0389	0.0261	0.0251
	20000	18	25	17	17	57	104	51	35	0.0530	0.0998	0.0495	0.0309
x_3	5000	19	22	21	18	56	72	66	38	0.0207	0.0278	0.0234	0.0134
	10000	19	24	19	20	56	80	57	44	0.0331	0.0430	0.0406	0.0206
	20000	19	25	21	17	55	83	65	35	0.0618	0.0814	0.0627	0.0309

Table 5 Numerical results of Problem 5

SP	DIM	NI				NFE				CPU			
		SASCGM	DLPM	MLS	SDYP	SASCGM	DLPM	MLS	SDYP	SASCGM	DLPM	MLS	SDYP
x_0	5000	8	6	6	6	17	13	13	13	0.0027	0.0022	0.0317	0.0016
	10000	7	6	6	6	15	13	13	13	0.0039	0.0039	0.0033	0.0024
	20000	7	5	5	5	15	11	11	11	0.0069	0.0055	0.0048	0.0034
x_1	5000	7	5	5	13	9	6	6	26	0.0019	0.0014	0.0013	0.0032
	10000	8	5	5	13	11	6	6	26	0.0040	0.0026	0.0021	0.0051
	20000	8	5	5	13	11	6	6	26	0.0070	0.0045	0.0039	0.0085
x_2	5000	17	14	14	10	35	29	29	19	0.0056	0.0053	0.0046	0.0024
	10000	17	14	14	10	35	29	29	19	0.0100	0.0094	0.0079	0.0040
	20000	18	14	14	10	37	29	29	19	0.0179	0.0161	0.0140	0.0063
x_3	5000	5	4	4	13	6	5	5	26	0.0014	0.0012	0.0011	0.0034
	10000	5	4	4	14	6	5	5	28	0.0024	0.0020	0.0018	0.0056
	20000	5	4	4	14	6	5	5	28	0.0042	0.0036	0.0031	0.0097

Table 6 Numerical results of Problem 6

SP	DIM	NI				NFE				CPU			
		SASCGM	DLPM	MLS	SDYP	SASCGM	DLPM	MLS	SDYP	SASCGM	DLPM	MLS	SDYP
x_0	5000	20	30	19	20	75	184	57	46	0.0177	0.0433	0.0207	0.0068
	10000	19	22	20	20	72	77	64	46	0.0228	0.0373	0.0203	0.0196
	20000	21	21	21	21	81	70	67	49	0.0572	0.0469	0.0594	0.0259
x_1	5000	23	26	21	21	85	90	66	46	0.0133	0.0169	0.0118	0.0069
	10000	23	24	27	24	86	81	82	58	0.0263	0.0295	0.0292	0.0228
	20000	22	28	23	23	87	143	78	52	0.0713	0.0951	0.0684	0.0413
x_2	5000	20	25	19	20	78	99	59	46	0.0120	0.0178	0.0106	0.0068
	10000	20	22	18	22	80	97	53	54	0.0302	0.0450	0.0187	0.0225
	20000	22	24	18	18	84	93	56	40	0.0694	0.0593	0.0429	0.0213
x_3	5000	21	29	22	23	77	150	68	53	0.0120	0.0272	0.0122	0.0122
	10000	21	24	21	21	80	80	65	47	0.0328	0.0275	0.0304	0.0130
	20000	22	22	24	23	90	70	72	53	0.0476	0.0455	0.0441	0.0265

Table 7 Numerical results of Problem 7

SP	DIM	NI				NFE				CPU			
		SASCGM	DLPM	MLS	SDYP	SASCGM	DLPM	MLS	SDYP	SASCGM	DLPM	MLS	SDYP
x_0	5000	3	5	5	5	11	16	16	12	0.0096	0.0166	0.0201	0.0147
	10000	2	5	5	5	7	16	16	12	0.0162	0.0284	0.0380	0.0205
	20000	7	4	4	4	27	13	13	10	0.1062	0.0535	0.0440	0.0331
x_1	5000	41	31	30	57	160	117	111	190	0.1409	0.1063	0.0994	0.1685
	10000	41	31	32	58	160	114	117	193	0.3351	0.2038	0.2106	0.3586
	20000	39	30	32	57	152	109	118	187	0.5987	0.3983	0.4295	0.7145
x_2	5000	38	30	28	55	148	112	103	183	0.1305	0.1064	0.0927	0.2220
	10000	38	27	30	55	148	99	109	182	0.2898	0.1833	0.2068	0.3617
	20000	36	28	30	54	140	103	108	176	0.5344	0.3519	0.4036	0.6686
x_3	5000	38	30	28	55	148	112	103	183	0.1394	0.1235	0.1278	0.1581
	10000	38	27	30	55	148	99	109	182	0.2931	0.1757	0.2005	0.3389
	20000	36	28	30	54	140	103	108	176	0.4699	0.3714	0.4239	0.5990

Table 8 Numerical results of Problem 8

SP	DIM	NI				NFE				CPU			
		SASCGM	DLPM	MLS	SDYP	SASCGM	DLPM	MLS	SDYP	SASCGM	DLPM	MLS	SDYP
x_0	5000	6	3	3	9	24	10	10	21	0.0026	0.0013	0.0189	0.0023
	10000	6	3	3	9	24	10	10	21	0.0042	0.0022	0.0021	0.0036
	20000	6	3	3	10	24	10	10	23	0.0071	0.0039	0.0035	0.0078
x_1	5000	7	4	4	10	28	13	13	26	0.0031	0.0018	0.0018	0.0027
	10000	7	5	5	11	28	17	17	28	0.0050	0.0040	0.0036	0.0047
	20000	7	5	5	11	28	17	17	28	0.0085	0.0070	0.0062	0.0077
x_2	5000	4	6	6	10	16	23	23	23	0.0018	0.0031	0.0029	0.0026
	10000	5	6	6	10	20	23	23	23	0.0036	0.0050	0.0046	0.0039
	20000	5	6	6	10	20	23	23	23	0.0059	0.0089	0.0080	0.0065
x_3	5000	6	4	4	10	24	13	13	24	0.0027	0.0018	0.0017	0.0027
	10000	7	4	4	10	28	13	13	24	0.0050	0.0030	0.0027	0.0040
	20000	7	4	4	10	28	14	14	24	0.0084	0.0055	0.0049	0.0067

Table 9 Numerical results of Problem 9

SP	DIM	NI				NFE				CPU			
		SASCGM	DLPM	MLS	SDYP	SASCGM	DLPM	MLS	SDYP	SASCGM	DLPM	MLS	SDYP
x_0	5000	8	11	11	10	27	33	33	21	0.0039	0.0056	0.0087	0.0029
	10000	9	11	11	10	31	33	33	21	0.0071	0.0094	0.0085	0.0045
	20000	9	11	11	11	31	33	33	23	0.0120	0.0166	0.0150	0.0085
x_1	5000	4	12	12	11	10	35	35	23	0.0016	0.0060	0.0054	0.0031
	10000	5	12	12	11	14	35	35	23	0.0034	0.0099	0.0089	0.0048
	20000	5	13	13	11	14	39	39	23	0.0058	0.0194	0.0172	0.0084
x_2	5000	4	11	11	9	12	34	34	19	0.0017	0.0055	0.0051	0.0025
	10000	4	11	11	9	12	34	34	19	0.0027	0.0089	0.0087	0.0038
	20000	4	11	11	9	12	34	34	19	0.0046	0.0166	0.0147	0.0068
x_3	5000	6	12	12	12	17	36	36	26	0.0022	0.0057	0.0052	0.0030
	10000	6	13	13	12	17	39	39	26	0.0041	0.0109	0.0094	0.0050
	20000	7	12	12	13	21	36	36	28	0.0085	0.0177	0.0157	0.0097

Table 10 Numerical results of Problem 10

SP	DIM	NI				NFE				CPU			
		SASCGM	DLPM	MLS	SDYP	SASCGM	DLPM	MLS	SDYP	SASCGM	DLPM	MLS	SDYP
x_0	5000	7	13	13	11	21	40	40	23	0.0027	0.0060	0.0092	0.0027
	10000	7	13	13	11	21	40	40	23	0.0043	0.0098	0.0087	0.0041
	20000	8	13	13	11	24	40	40	23	0.0082	0.0168	0.0150	0.0065
x_1	5000	6	12	12	6	18	37	37	12	0.0021	0.0052	0.0046	0.0013
	10000	6	13	13	6	18	40	40	12	0.0036	0.0098	0.0087	0.0022
	20000	6	13	13	7	18	40	40	14	0.0060	0.0170	0.0149	0.0042
x_2	5000	6	10	10	7	16	29	29	14	0.0020	0.0044	0.0037	0.0017
	10000	6	10	10	7	16	29	29	14	0.0034	0.0071	0.0064	0.0024
	20000	6	14	14	7	16	42	42	14	0.0057	0.0179	0.0155	0.0040
x_3	5000	5	9	9	7	15	28	28	15	0.0019	0.0040	0.0036	0.0015
	10000	5	9	9	7	15	28	28	15	0.0029	0.0065	0.0057	0.0026
	20000	5	10	10	7	15	31	31	15	0.0047	0.0129	0.0112	0.0043

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

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

Problem 3. Liu and Feng [18] The mapping F is given by

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

Problem 4. Liu and Li [20] The mapping F is given by

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

Problem 5. Abubakar and Kumam [21] The mapping F is given by

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

Problem 6. Hu and Wei [11] The mapping F is given by

$$\begin{aligned} F_1(x) &= 2.5x_1 + x_2 - 1, \\ F_i(x) &= x_{i-1} + 2.5x_i + x_{i+1} - 1, \quad i = 2, 3, \ldots, n-1, \\ F_n(x) &= x_{n-1} + 2.5x_n - 1. \end{aligned}$$

Problem 7. Hu and Wei [11] The mapping F is given by

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

where $h = \frac{1}{n+1}$.

Problem 8. Wang and Guan [25] The mapping F is given by

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

Problem 9. Wang and Guan [25] The mapping F is given by

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

Problem 10. Gao and He [24] The mapping F is given by

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

The numerical results are reported in Tables 1, 2, 3, 4, 5, 6, 7, 8, 9, and 10, where "SP" represents the starting point (initial point), "DIM" denotes the dimension of the problem, "NI" refers to the number of iterations, "NFE" stands for the number of function evaluations, and "CPU" is the CPU time in seconds. In Table 3, "*" indicates that the algorithm did not converge within the maximum number of iterations. From the tables, we observe

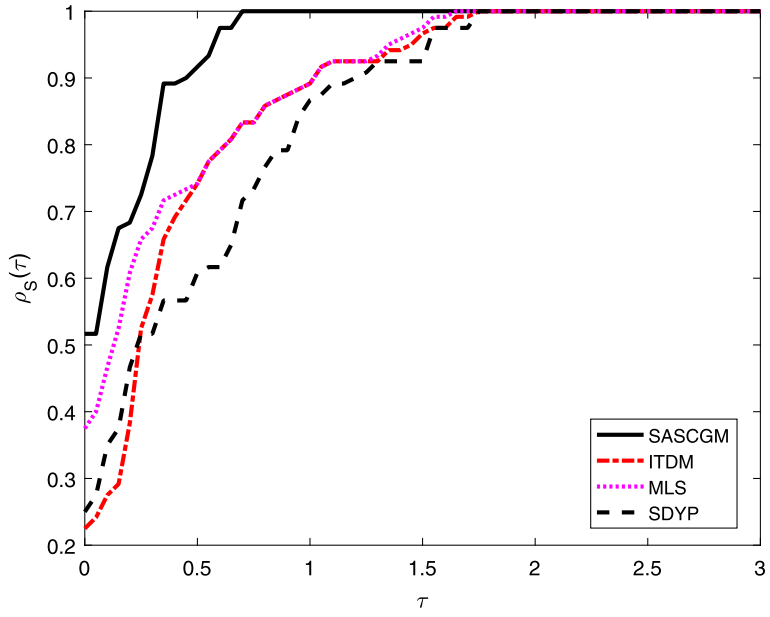

Fig. 1 Iterations performance profile

that the proposed method performs better than the other methods in Problems 2, 3, 4, 6, 9, and 10. The proposed method performs slightly lower in Problems 1, 5, 7, and 8. However, overall, the proposed method shows that it is very competitive with the other methods and can be a good addition to the existing methods in the literature.

The performance of the three methods is further presented graphically in Figs. 1, 2 , and 3 based on the number of iterations (NI), number of function evaluations (NFE), and the CPU time, respectively, using the performance profile of Dolan and Moré [28]. That is, we

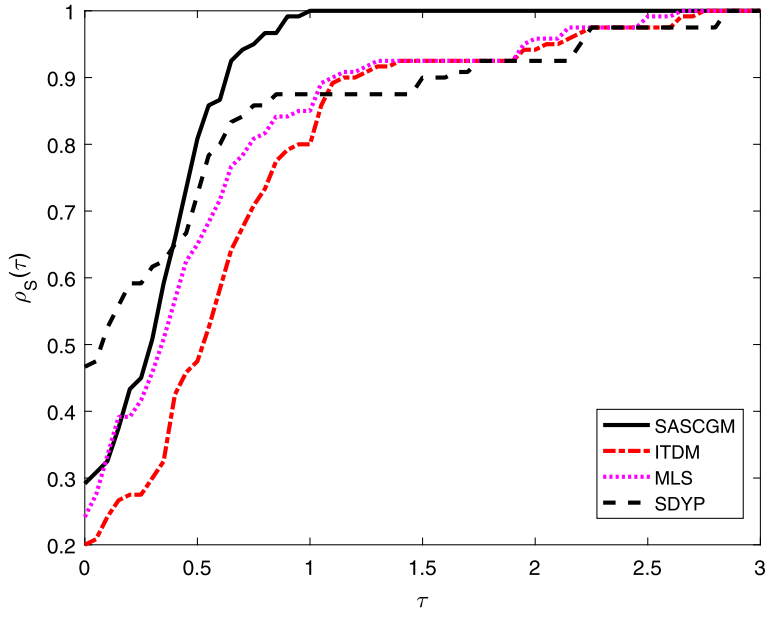

Fig. 2 Function evaluations performance profile

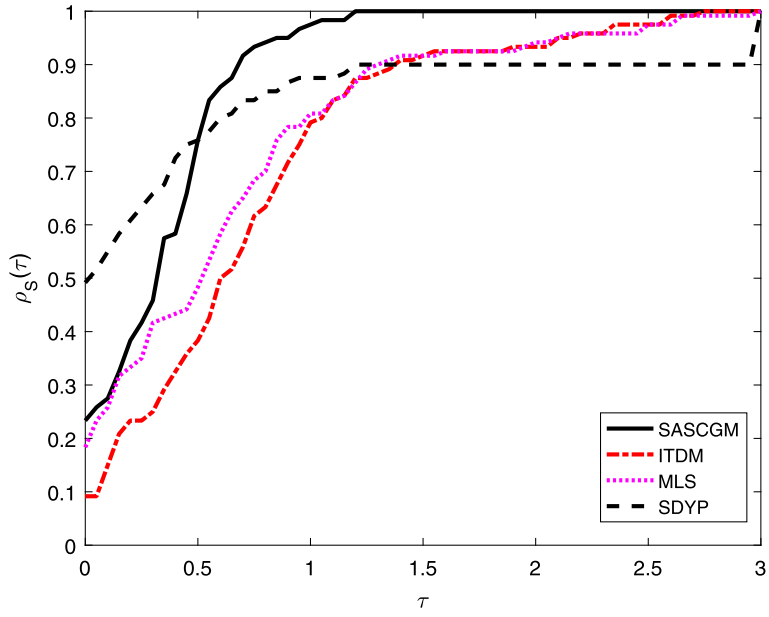

Fig. 3 Cpu time performance profile

plot the probability $\rho_S(\tau)$ of the test problems for which each of the three methods was within a factor τ. Figures 1, 2, and 3 clearly show the efficiency of the proposed *SASCGM* method as compared to the other three methods.

Conclusion

In this paper, we proposed a self adaptive spectral conjugate gradient-based projection (SASCGM) method for solving systems of large-scale nonlinear monotone equations. The proposed method is free from derivative evaluations and also satisfies the descent condition $F_k^T d_k \leq -c\|F_k\|^2, c > 0$, independent of any line search. The global convergence of the proposed method was also established. The proposed algorithm was tested on some benchmark problems with different initial points and different dimensions and the numerical results show that the method is competitive.

Abbreviations

CPU: CPU time is seconds; DIM: Dimension; ITDM: Improved three-term derivative-free method; MLS: Modified Liu-Storey method; NFE: Number of function evaluations; NI: Number of iterations; SASCGM: Self adaptive spectral conjugate gradient-based projection method; SDYP: Spectral DY-type projection method; SP: Starting (initial) point

Acknowledgements

The authors appreciate the work of the referees, for their valuable comments and suggestions that led to the improvement of this paper. The authors would also like to thank Prof J. Liu who provided the MATLAB code for SDYP.

Authors' contributions

The authors jointly worked on the results, read, and approved the final manuscript.

Funding

None.

Availability of data and materials

All data generated or analyzed during this study are included in this manuscript.

Competing interests

The authors declare that they have no competing interests.

Received: 29 May 2019 Accepted: 24 December 2019
Published online: 13 January 2020

References

1. Meintjes, K., Morgan, A. P.: A methodology for solving chemical equilibrium systems. Appl. Math. Comput. **22**, 333–361 (1987)
2. Dirkse, S. P., Ferris, M. C.: A collection of nonlinear mixed complementarity problems. Optim. Methods Softw. **5**, 319–345 (1995)
3. Zhao, Y. B., Li, D.: Monotonicity of fixed point and normal mapping associated with variational inequality and its applications. SIAM J. Optim. **11**, 962–973 (2001)
4. Dehghan, M., Hajarian, M.: New iterative method for solving non-linear equations with fourth-order convergence. Int. J. Comput. Math. **87**, 834–839 (2010)
5. Li, D., Fukushima, M.: A global and superlinear convergent Gauss-Newton-based BFGS method for symmetric nonlinear equations. SIAM J. Numer. Anal. **37**, 152–172 (1999)
6. Dehghan, M., Hajarian, M.: On some cubic convergence iterative formulae without derivatives for solving nonlinear equations. Int. J. Numer. Methods Biomed. Eng. **27**, 722–731 (2011)
7. Dehghan, M., Hajarian, M.: Some derivative free quadratic and cubic convergence iterative formulas for solving nonlinear equations. Comput. Appl. Math. **29**, 19–30 (2010)
8. Zhou, G., Toh, K. C.: Superlinear convergence of a Newton-type algorithm for monotone equations. J. Optim. Theory Appl. **125**, 205–221 (2005)
9. Zhou, W. J., Liu, D. H.: A globally convergent BFGS method for nonlinear monotone equations without any merit functions. Math. Comput. **77**, 2231–2240 (2008)
10. Solodov, M. V., Svaiter, B. F.: A globally convergent inexact newton method for systems of monotone equations, Reformulation: Nonsmooth, Piecewise Smooth, Semismoothing methods. Springer US (1998). https://doi.org/10.1007/978-1-4757-6388-1_18
11. Hu, Y., Wei, Z.: A modified Liu-Storey conjugate gradient projection algorithm for nonlinear monotone equations. Int. Math. Forum. **9**, 1767–1777 (2014)
12. Feng, D., Sun, M., Wang, X.: A family of conjugate gradient methods for large scale nonlinear equations. J. Inqual. Appl. **2017**, 236 (2017)
13. Ding, Y., Xiao, Y., Li, J.: A class of conjugate gradient methods for convex constrained monotone equations. Optim. **66**(12), 2309–2328 (2017)
14. Koorapetse, M., Kaelo, P.: Globally convergent three-term conjugate gradient projection methods for solving nonlinear monotone equations. Arab. J. Math. **7**, 289–301 (2018)
15. Liu, J., Li, S.: Multivariate spectral DY-type projection method for convex constrained nonlinear monotone equations. J. Ind. Manag. Optim. **13**, 283–295 (2017)
16. Wang, X. Y., Li, S. J., Kou, X. P.: A self-adaptive three-term conjugate gradient method for monotone nonlinear equations with convex constraints. Calcolo. **53**, 133–145 (2016)
17. Xiao, Y., Zhu, H.: A conjugate gradient method to solve convex constrained monotone equations with applications in compressive sensing. J. Math. Anal. Appl. **405**, 310–319 (2013)
18. Liu, J., Feng, Y.: A derivative-free iterative method for nonlinear monotone equations with convex constraints. Numer. Algor. **82**, 245–262 (2019)
19. Sun, M., Liu, J.: Three derivative-free projection methods for nonlinear equations with convex constraints. J. Appl. Math. Comput. **47**, 265–276 (2015)
20. Liu, J., Li, S.: A three-term derivative-free projection method for nonlinear monotone system of equations. Calcolo. **53**, 427–450 (2016)
21. Abubakar, A. B., Kumam, P.: An improved three-term derivative-free method for solving nonlinear equations. Comput. Appl. Math. **37**(5), 6760–6773 (2018)
22. Liu, J., Li, S.: Spectral DY-type projection method for nonlinear monotone systems of equations. J. Comput. Math. **33**, 341–354 (2015)
23. Abubakar, A. B., Kumam, P.: A descent Dai-Liao conjugate gradient method for nonlinear equations. Numer. Algor. **81**, 197–210 (2019)
24. Gao, P., He, C.: An efficient three-term conjugate gradient method for nonlinear monotone equations with convex constraints. Calcolo. **55**, 53 (2018). https://doi.org/10.1007/s10092-018-0291-2
25. Wang, S, Guan, H: A scaled conjugate gradient method for solving monotone nonlinear equations with convex constraints. J. Appl. Math. **2013**, 286486 (2013)
26. Ou, Y., Li, J.: A new derivative-free SCG-type projection method for nonlinear monotone equations with convex constraints. J. Appl. Math. Comput. **56**, 195–216 (2018)
27. Yuan, G., Zhang, M.: A three-terms Polak-Ribiere-Polyak conjugate gradient algorithm for large-scale nonlinear equations. J. Comput. Appl. Math. **286**, 186–195 (2015)
28. Dolan, E. D., More, J. J.: Benchmarking optimization software with performance profiles. Math. Program. **91**, 201–213 (2002)

Publisher's Note