CompAMa

www.compama.co.usb.ve

Bull. Comput. Appl. Math., Vol.5, No.1, 2017

ISSN 2244-8659

A positive spectral gradient-like method for large-scale nonlinear monotone equations

Hassan Mohammad [1], *Auwal Bala Abubakar* [2]

CompAMa Vol.5, No.1, pp.99-115, 2017 - Accepted May 26, 2017

Abstract

In this work, we proposed a combine form of a positive spectral gradient-like method and projection method for solving nonlinear monotone equations. The spectral gradient-like coefficient is obtained using a convex combination of two different positive spectral coefficients. Under the monotonicity and Lipschitz continuity assumptions, the method is shown to be globally convergent. We show the numerical efficiency of the method by comparing it with the existing methods.

Keywords: Non-linear equations, monotone equations, spectral gradient method, projection method.

1 Introduction

Consider the problem of solving nonlinear system of equations

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

where $F : \mathbb{R}^n \to \mathbb{R}^n$ is continuous and monotone. That is

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

[1]Departament of Mathematical Sciences, Faculty of Sciences, Bayero University, Kano, Nigeria (hmuhd.mth@buk.edu.ng or hmuhammad.mth@buk.edu.ng).

[2]Departament of Mathematical Sciences, Faculty of Sciences, Bayero University, Kano, Nigeria (ababubakar.mth@buk.edu.ng).

Systems of monotone equations have many practical backgrounds such as the first-order necessary condition of the unconstrained convex optimization problem [1] and the subproblems in the generalized proximal algorithm with Bregman distances [2]. Some monotone nonlinear complementarity problems and variational inequality problems can be transformed into monotone nonlinear equations [3, 4].

Among the basic iterative methods for solving (1) includes Newton's method, Quasi-Newton methods, and variants (see for example, [5–13]). These methods are attractive because of their fast convergence rates. However, they are not suitable for solving monotone equations because they require computation of Jacobian matrix or approximation of it, and solving a linear system of equations in every iteration.

Solodov and Svaiter [14], combine Newton's method and projection strategy to develop a global convergent inexact Newton method for system of monotone equations. A remarkable property of the method is that without the regularity assumption, the whole sequence of iterates converges to a solution of the system. Zhou and Toh [15] improved the work by Solodov and Svaiter and obtained a Newton-type method with superlinear convergence. A quasi-Newton's method that employ the projection strategy was presented by Zhou and Li [16]. Zhang and Zhou [17] proposed a combination of the spectral gradient method [18] with the projection method. Their method is globally convergent provided that the nonlinear equations to be solved are monotone and Lipschitz continuous. Iterative methods for solving monotone equations are now receiving more attention, just recently, La Cruz [19] proposed a variant of the so called DF-SANE [20] method for solving large-scale monotone equations.

Inspired by the above developments and due to the simple implementation, some conjugate gradient based methods for large-scale systems of monotone equations have been introduced, see for example [21–28] and reference therein.

Basically, iterative scheme for solving (1) has the general form: given the initial approximation x_0, a sequence of iterates $\{x_k\}$ is obtained via

$$x_{k+1} = x_k + s_k, \tag{3}$$

where $s_k = \alpha_k d_k$, α_k is the step length obtained by a suitable line search and

d_k is the search direction which is given by

$$d_k = \begin{cases} -F(x_k), & \text{if } k = 0, \\ -\theta_k F(x_k) + \beta_k d_{k-1}, & \text{if } k \geq 1, \end{cases} \tag{4}$$

where θ_k and β_k are parameters which are to be determine.

If $\theta_k = 1$ and $\beta_k \neq 0$ in (4), then we get the classical conjugate gradient algorithms according to the value of the parameter β_k. Else if $\beta_k = 0$, then another class of algorithm is obtained according to the choice of θ_k. Either θ_k is a positive scalar or a square matrix. If $\theta_k = 1$ we have the steepest descent algorithm. If $\theta_k = (F'(x_k))^{-1}$ the inverse Jacobian matrix or an approximation of it, then we get the Newton or the quasi-Newton algorithms, respectively. A special case in which

$$\theta_k = \lambda_k I, \text{ where } \lambda_k = \frac{s_{k-1}^T s_{k-1}}{y_{k-1}^T s_{k-1}}, \; y_{k-1} = F(x_k) - F(x_{k-1}), \tag{5}$$

I is the identity matrix and λ_k corresponding to the inverse of the Rayleigh quotient is the spectral gradient (or Barzilai and Borwien) method. Therefore we can see that in general case, when $\theta_k \neq 0$ is selected in a quasi-Newton manner and $\beta_k \neq 0$, then equation (4) is a combination of conjugate gradient method and quasi-Newton methods.

Motivated by the work of Zhang and Zhou [17] and the positive Barzilai-Borwein-like step-size used by Dai et al. [29] to solved symmetric linear systems, we present a modified positive spectral coefficient which is the convex combination of the default spectral coefficient [18] and the positive spectral coefficient [29]. The remarkable future of our approach is that the spectral coefficient is always positive and if F is a gradient vector of a real-valued function $f : \mathbb{R}^n \to \mathbb{R}$ then the direction is a sufficiently descent direction of f at x_k.

The remaining part of this paper is organized as follows. In section 2 we described the proposed method and its algorithm. The global convergence is established in section 3 and numerical results are reported in section 4. Throughout this paper $\|.\|$ stands for the Euclidean norm.

2 Algorithm

In this section, we first recall the spectral gradient method for unconstrained optimization by Barzilai and Borwein [18]. In this method the iterative sequence is obtained via

$$x_{k+1} = x_k - \lambda_k g_k, \tag{6}$$

where λ_k is given in (5) and g_k is the gradient of the function $f : \mathbb{R}^n \to \mathbb{R}$.

Secondly, we recall the hyperplane projection method by Solodov and Svaiter [14]. Let x_k be the current iterate, by performing some kind of line search procedure along the direction d_k a point $z_k = x_k + \alpha_k d_k$ can be computed such that

$$F(z_k)^T(x_k - z_k) > 0.$$

By the monotonicity of F,

$$F(z_k)^T(x^* - z_k) \leq 0,$$

for all x^* such that $F(x^*) = 0$.

It follows that, the hyperplane

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

strictly separates x_k from the solution set of Equation (1).

Once the separating hyperplane is constructed, the next iterate x_{k+1} can be computed by projecting x_k onto it. That is,

$$x_{k+1} = x_k - \frac{F(z_k)^T(x_k - z_k)F(z_k)}{\|F(z_k)\|^2}. \tag{7}$$

We now formally present our algorithm as follows:

Algorithm 1 (PSG)

Step 0. Given $x_0 \in D \subset \mathbb{R}^n, \beta, \sigma, \tau \in (0,1)$, stopping tolerance $\epsilon > 0$, Set $k = 0$.

Step 1. Compute $F(x_k)$. If $\|F(x_k)\| \leq \epsilon$ stop.

Step 2. Compute $d_k = -\lambda_k F(x_k), \quad d_0 = -F(x_0)$, where

$$\lambda_k = (1 - \tau)\theta_k^* + \tau\theta_k^{**}, \tag{8}$$

where

$$\theta_k^* = \frac{s_{k-1}^T s_{k-1}}{y_{k-1}^T s_{k-1}}, \quad \theta_k^{**} = \frac{\|s_{k-1}\|}{\|y_{k-1}\|},\ y_{k-1} = F(x_k) - F(x_{k-1}) + r_k s_{k-1},\ r_k = \frac{1}{(k+1)^2}.$$

Stop if $d_k = 0$.

Step 3. Determine $\alpha_k = \beta^{m_k}$ with m_k being the smallest nonnegative integer m such that

$$-F(x_k + \beta^m d_k)^T d_k \geq \sigma \beta^m \|F(x_k + \beta^m d_k)\| \|d_k\|^2. \tag{9}$$

Step 4. Compute $z_k = x_k + \alpha_k d_k$. If $\|F(z_k)\| = 0$ stop.

Step 5. Compute x_{k+1} using Equation (7).

Step 6. Let $k = k + 1$ and go to **Step 1**.

Remarks

1. In Step 2 of Algorithm 1, the definition of y_{k-1} is different from the one given in (5) and by the monotonicity of F, it is not difficult to see that $y_{k-1}^T s_{k-1} > 0$.

2. The spectral coefficient λ_k is positive $\forall k \in \mathbb{N} \cup \{0\}$.

3. Since $F(x_k)^T d_k \leq -c\|F(x_k)\|^2$. $\forall k, \quad c > 0$, it is clear that the line search (9) holds for all sufficiently small $\alpha_k > 0$. Therefore, Algorithm 1 is well-defined.

3 Convergence Results

In order to prove the global convergence results of Algorithm 1, we need the following preliminaries:

Lemma 1. *[14] Suppose that $x^* \in \mathbb{R}^n$ satisfies $F(x^*) = 0$. Let $\{x_k\}$ be generated by Algorithm 1. Then*

$$\|x_{k+1} - x^*\|^2 \leq \|x_k - x^*\|^2 - \|x_{k+1} - x_k\|^2. \tag{10}$$

Lemma 2. *Suppose F is Lipschitz continuous. Let $\{d_k\}$ be the sequence of directions generated by Algorithm 1, then there exists a constant $M > 0$ such that $\|d_k\| \leq M \quad \forall k \in \mathbb{N} \cup \{0\}$.*

Proof. Lemma 1 implies that the sequence $\{\|x_k - x^*\|\}$ is non-increasing and convergent, therefore bounded. Also, $\{x_k\}$ is bounded and

$$\lim_{k\to\infty} \|x_{k+1} - x_k\| = 0. \tag{11}$$

From (7) and line search (9)

$$\begin{aligned} \|x_{k+1} - x_k\| &= \frac{|F(z_k)^T(x_k - z_k)|}{\|F(z_k)\|^2}\|F(z_k)\| \\ &= \frac{|\alpha_k F(z_k)^T d_k|}{\|F(z_k)\|} \\ &\geq \frac{\sigma\alpha_k^2\|F(z_k)\|\|d_k\|^2}{\|F(z_k)\|} \\ &= \sigma\alpha_k^2\|d_k\|^2 \geq 0. \end{aligned} \tag{12}$$

By (11) and (12) it follows that

$$\lim_{k\to\infty} \alpha_k\|d_k\| = 0. \tag{13}$$

Let x^* be any point in $\mathbb{R}^n$ such that $F(x^*) = 0$, Lemma 1 implies $\|x_k - x^*\| \leq \|x_0 - x^*\|$.
Now, since F is Lipschitz continuous,

$$\begin{aligned} \|F(x_k)\| &= \|F(x_k) - F(x^*)\| \\ &\leq L\|x_k - x^*\| \\ &\leq L\|x_0 - x^*\|. \end{aligned}$$

Taking $M_1 := L\|x_0 - x^*\|$, we have $\|F(x_k)\| \leq M_1, \quad \forall k \in \mathbb{N}\cup\{0\}$. From Step 2 of Algorithm 1,

$$\|d_0\| = \|F(x_0)\| \leq M_1,$$

and

$$\begin{aligned} \|d_k\| &= |\lambda_k|\|F(x_k)\| \\ &\leq |\lambda_k|M_1 \end{aligned}$$

Equation(13) implies, there exists a positive integer k_0 such that

$$\alpha_{k-1}\|d_{k-1}\| \leq \epsilon_0, \quad \forall k > k_0,$$

for an arbitrary constant ϵ_0. Taking $M = \max\{\|d_0\|, \|d_1\|, ..., \|d_{k_0}\|, |\lambda_k|M_1\}$, we have

$$\|d_k\| \leq M \quad \forall k \in \mathbb{N} \cup \{0\}.$$

□

The following theorem establish the global convergence of Algorithm 1.

Theorem 1. *Let $\{x_k\}$ and $\{z_k\}$ be sequences generated by Algorithm 1. Then*

$$\lim_{k\to\infty} \inf \|F(x_k)\| = 0. \tag{14}$$

Proof. If $\liminf_{k\to\infty} \|d_k\| = 0$, we have $\liminf_{k\to\infty} \|F(x_k)\| = 0$. By continuity of F, the sequence $\{x_k\}$ has some accumulation point $\tilde{x}$ such that $F(\tilde{x}) = 0$. Since $\{\|x_k - \tilde{x}\|\}$ converges and $\tilde{x}$ is an accumulation point of $\{x_k\}$ it follows that $\{x_k\}$ converges to $\tilde{x}$.

If $\liminf_{k\to\infty} \|d_k\| > 0$, we have $\liminf_{k\to\infty} \|F(x_k)\| > 0$. By (13), it holds that $\lim_{k\to\infty} \alpha_k = 0$.

From the line search (9),

$$-F(x_k + \beta^{m_{k-1}} d_k)^T d_k < \sigma \beta^{m_{k-1}} \|F(x_k + \beta^{m_{k-1}} d_k)\| \|d_k\|^2.$$

Using the boundedness of $\{x_k\}, \{d_k\}$, we can choose a subsequence such that allowing k to go to infinity in the above inequality results

$$F(\tilde{x})^T \tilde{d} > 0. \tag{15}$$

On the other hand, allowing k to approach ∞ in (9), implies

$$F(\tilde{x})^T \tilde{d} \leq 0. \tag{16}$$

Therefore, (15) and (16) cannot hold concurrently. Hence, it is not possible to have $\liminf_{k\to\infty} \|F(x_k)\| > 0$ and the proof is complete. □

4 Numerical Results

In this section, we perform some numerical experiment to investigate the efficiency of the proposed method. All algorithms were implemented using MATLAB R2010a and run on a PC with Intel COREi5 processor with 4GB of RAM and CPU 2.3GHZ. We test problems 1 to 10 with different initial points $X1 = (1, 1, ..., 1)$, $X2 = (-1, -1, ..., -1)$, $X3 = (-0.1, -0.1, ..., 0.1)$, $X4 = (0.1, 0.1, ..., 0.1)$, $X5 = (1, \frac{1}{2}, ..., \frac{1}{n})$, $X6 = (1 - \frac{1}{n}, 2 - \frac{2}{n}, ..., 0)$, $X7 = (10, 10, ..., 10)$ and $X8 = (-10, -10, ..., -10)$. For problems 1 to 8 we used uniform dimensions ranging from 1000 to 10000, while we used perfect square dimensions ranging from 900 to 12100 for the rest of the problems. In our experiment we use the symbol $'-'$ to report the failure of a method when the number of iterations is greater than or equal to 1000.

In PSG algorithm, we set $\sigma = 0.01, \beta = 0.8, \tau = \frac{1}{e^{(k+1)^{(k+1)}}}$, except for problem 7 where we set $\beta = 0.6$ for the initial point $X8$. In SDYP and SP algorithms, we set $\sigma = 0.01, \beta = 0.5, 0.4$ respectively (as originally given in the papers). All runs were stop whenever $\|F_k\| < 10^{-4}$.

In Table 1 and Table 2 we present results on the following information: the number of iterations (ITER) needed to converge to an approximate solution, the CPU time in seconds (TIME), the number of function evaluation (FEVAL) and the norm of the objective function F at the approximate solution x^* (NORM). The acronym 'NaN' appearing in Table 2 means 'Not a Number'.

In addition, Table 3 summarized the results obtained from Table 1 and Table 2 based on which each method is a winner in terms of CPU time (TIME), number of iterations (ITER) and number of function evaluations (FEVAL).

It can be observed from Table 3 that PSG method solved about 60% of the total test problems within a shorter time than SP (8.75%) and SDYP (17.5%). In terms of number of iterations PSG is the most efficient because it solved about 62.5% with less number of iterations than SP (0%) and SDYP (27.5%). It is worth mentioning that our proposed PSG method solved the last two test functions (arise from the discretization of some differential equations) successfully, while SP and SDYP failed to solved those function within the maximum number of iterations required.

The test functions

$$F(x) = (f_1(x), f_2(x), ..., f_n(x))^T, \quad \text{where} \quad x = (x_1, x_2, ..., x_n)^T$$

are listed as follows:

Problem 1 [30]

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

Tab. 1: Numerical Results for SP, SDYP and PSG methods for Problem 1 to 5

	Initial	**n**	**ITER**			**TIME**			**Feval**			**Norm**		
Problems			**SP**	**SDYP**	**PSG**	**SP**	**SDYP**	**PSG**	**SP**	**SDYP**	**PSG**	**SP**	**SDYP**	**PSG**
	X1	**1000**	612	562	306	0.977096	0.347874	0.219974	1836	1686	918	9.98E-05	1.00E-04	9.94E-05
	X2	**1000**	23	7	9	0.015147	0.010177	0.009076	69	21	27	7.39E-05	9.58E-05	9.42E-05
	X3	**1000**	18	6	9	0.009932	0.006993	0.005697	54	18	27	7.87E-05	3.52E-05	3.21E-05
	X4	**1000**	593	548	306	0.264398	0.258694	0.157252	1779	1664	918	1.00E-04	9.99E-05	9.94E-05
	X5	**10000**	308	289	298	1.182605	1.051789	1.081876	924	867	894	9.96E-05	9.97E-05	9.99E-05
P1	**X6**	**10000**	-	-	483	-	-	1.74165	-	-	1449	-	-	9.95E-05
	X7	**10000**	-	-	483	-	-	1.847287	-	-	1449	-	-	9.96E-05
	X8	**10000**	40	19	18	0.212785	0.175317	0.197806	120	57	54	6.98E-05	5.77E-05	9.30E-05
	X1	**1000**	25	9	10	0.01616	0.00664	0.005848	75	27	30	6.96E-05	8.16E-05	6.82E-05
	X2	**1000**	24	10	9	0.013979	0.009998	0.007857	72	30	27	9.55E-05	7.42E-05	7.68E-05
	X3	**1000**	20	7	7	0.010755	0.007575	0.004995	60	21	21	7.66E-05	4.35E-06	4.89E-05
	X4	**1000**	20	8	9	0.010505	0.01801	0.005267	60	24	27	8.18E-05	4.32E-05	8.25E-05
	X5	**10000**	20	16	14	0.075477	0.078249	0.090461	60	48	42	7.47E-05	7.03E-05	6.61E-05
P2	**X6**	**10000**	27	22	14	0.104481	0.109667	0.076451	81	66	42	7.68E-05	8.01E-05	8.28E-05
	X7	**10000**	42	24	20	0.184385	0.178531	0.188336	126	72	60	6.68E-05	7.40E-06	6.56E-05
	X8	**10000**	40	32	19	0.180432	0.271937	0.169925	120	96	57	6.13E-05	7.51E-05	2.62E-05
	X1	**1000**	38	28	22	0.027194	0.030577	0.016944	114	84	66	7.15E-05	7.40E-05	4.79E-05
	X2	**1000**	41	29	26	0.025443	0.030163	0.020255	123	87	78	7.73E-05	8.11E-05	5.79E-05
	X3	**1000**	31	32	26	0.022175	0.032685	0.019942	93	96	78	7.34E-05	8.09E-05	5.62E-05
	X4	**1000**	36	30	23	0.023474	0.030629	0.018445	108	90	69	6.44E-05	9.40E-05	6.49E-05
	X5	**10000**	41	31	30	0.184813	0.245373	0.169784	123	93	90	6.82E-05	8.95E-05	5.78E-05
P3	**X6**	**10000**	43	29	-	0.18704	0.212561	-	129	87	-	7.99E-05	8.84E-05	-
	X7	**10000**	-	-	-	-	-	-	-	-	-	-	-	-
	X8	**10000**	-	-	-	-	-	-	-	-	-	-	-	-
	X1	**1000**	101	93	33	0.066758	0.085708	0.053158	303	279	99	9.74E-05	9.76E-05	5.21E-05
	X2	**1000**	64	87	42	0.040698	0.079832	0.060056	192	261	126	9.80E-05	9.57E-05	8.76E-05
	X3	**1000**	76	18	22	0.047969	0.018724	0.015137	228	54	66	9.35E-05	8.57E-05	2.71E-05
	X4	**1000**	71	59	24	0.046677	0.054305	0.031464	213	177	72	9.66E-05	9.00E-05	8.17E-05
	X5	**10000**	24	19	18	0.11024	0.13139	0.095178	72	57	54	8.94E-05	4.10E-05	8.79E-05
P4	**X6**	**10000**	103	92	41	0.479626	0.645417	0.499426	309	276	123	9.13E-05	9.11E-05	9.36E-05
	X7	**10000**	127	85	151	0.593355	0.692414	2.212888	381	255	453	9.81E-05	9.69E-05	6.36E-05
	X8	**10000**	94	85	148	0.404474	0.648166	2.212487	282	255	444	9.83E-05	9.55E-05	7.50E-05
	X1	**1000**	26	12	11	0.024195	0.015785	0.009507	78	36	33	6.57E-05	6.33E-06	6.38E-05
	X2	**1000**	27	12	12	0.023428	0.012696	0.009877	81	36	36	8.54E-05	1.32E-05	2.87E-05
	X3	**1000**	27	12	12	0.022585	0.012781	0.010076	81	36	36	6.47E-05	1.02E-05	2.18E-05
	X4	**1000**	27	12	11	0.023546	0.013085	0.009114	81	36	33	6.01E-05	9.48E-06	9.72E-05
	X5	**10000**	29	11	13	0.169439	0.091871	0.094327	87	33	39	9.65E-05	6.83E-05	4.59E-05
P5	**X6**	**10000**	28	12	13	0.166489	0.116191	0.09004	84	36	39	7.32E-05	1.23E-05	2.11E-05
	X7	**10000**	37	19	17	0.246461	0.182903	0.193747	111	57	51	6.34E-05	4.46E-05	5.52E-05
	X8	**10000**	45	27	23	0.33718	0.303648	0.351447	135	81	69	9.14E-05	4.44E-05	4.07E-05

Problem 2 [16]

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

Problem 3 [16]

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

Tab. 2: Numerical Results for SP, SDYP and PSG methods for Problem 6 to 10

	Initial	n	ITER			TIME			Feval			Norm		
Problems			**SP**	**SDYP**	**PSG**	**SP**	**SDYP**	**PSG**	**SP**	**SDYP**	**PSG**	**SP**	**SDYP**	**PSG**
	X1	**1000**	229	7	522	0.547922	0.050029	2.721665	687	21	1566	9.37E-05	NaN	9.32E-05
	X2	**1000**	235	7	-	0.474068	0.049591	-	705	21	-	9.76E-05	NaN	-
	X3	**1000**	112	66	142	0.21367	0.190808	0.533069	336	198	426	5.57E-05	6.22E-05	8.61E-05
	X4	**1000**	108	54	131	0.205038	0.158515	0.46199	324	162	393	6.95E-05	4.27E-05	8.72E-05
	X5	**10000**	-	-	-	-	-	-	-	-	-	-	-	-
P6	**X6**	**10000**	-	5	-	-	0.225795	-	-	15	-	-	NaN	-
	X7	**10000**	-	5	948	-	0.228469	35.70947	-	15	2844	-	NaN	8.35E-05
	X8	**10000**	-	5	-	-	0.223426		-	15	-	-	NaN	-
	X1	**1000**	54	39	26	0.703216	0.42539	0.268951	162	117	78	9.16E-05	7.87E-05	7.39E-05
	X2	**1000**	54	33	27	0.375009	0.34951	0.335551	162	99	81	9.23E-05	8.97E-05	7.06E-05
	X3	**1000**	47	28	20	0.331214	0.290744	0.184939	141	84	60	8.50E-05	8.67E-05	3.38E-05
	X4	**1000**	48	28	15	0.331568	0.29609	1.39E-01	144	84	45	9.21E-05	9.00E-05	6.61E-05
	X5	**10000**	39	31	16	1.90404	2.375007	0.978444	117	93	48	8.91E-05	8.64E-05	7.32E-05
P7	**X6**	**10000**	64	46	33	3.100786	3.891138	2.67857	192	138	99	9.22E-05	7.37E-05	9.01E-05
	X7	**10000**	89	4	816	4.256329	1.522045	115.7949	267	12	2448	8.53E-05	NaN	3.51E-05
	X8	**10000**	375	521	340	32.98647	72.32298	37.89541	1125	1563	1020	9.89E-05	8.27E-05	5.04E-05
	X1	**1000**	-	-	-	-	-	-	-	-	-	-	-	-
	X2	**1000**	-	-	-	-	-	-	-	-	-	-	-	-
	X3	**1000**	-	-	-	-	-	-	-	-	-	-	-	-
	X4	**1000**	-	-	-	-	-	-	-	-	-	-	-	-
	X5	**10000**	-	-	-	-	-	-	-	-	-	-	-	-
P8	**X6**	**10000**	-	-	-	-	-	-	-	-	-	-	-	-
	X7	**10000**	-	-	-	-	-	-	-	-	-	-	-	-
	X8	**10000**	-	-	-	-	-	-	-	-	-	-	-	-
	X1	**900**	-	-	135	-	-	3.315385	-	-	405	-	-	9.21E-05
	X2	**1600**	-	-	136	-	-	4.174302	-	-	408	-	-	9.51E-05
	X3	**2500**	-	-	133	-	-	4.324865	-	-	333	-	-	9.98E-05
	X4	**3600**	-	-	113	-	-	6.170219	-	-	339	-	-	9.61E-05
	X5	**4900**	-	-	126	-	-	8.498605	-	-	378	-	-	9.41E-05
P9	**X6**	**6400**	-	-	138	-	-	11.47546	-	-	414	-	-	9.37E-05
	X7	**8100**	-	-	158	-	-	16.20509	-	-	474	-	-	9.47E-05
	X8	**12100**	-	-	160	-	-	23.55938	-	-	480	-	-	9.32E-05
	X1	**900**	-	-	135	-	-	2.743291	-	-	405	-	-	9.22E-05
	X2	**1600**	-	-	136	-	-	3.1633	-	-	408	-	-	9.56E-05
	X3	**2500**	-	-	112	-	-	3.039852	-	-	336	-	-	9.28E-05
	X4	**3600**	-	-	113	-	-	4.136995	-	-	339	-	-	9.50E-05
	X5	**4900**	-	-	126	-	-	5.408373	-	-	378	-	-	9.54E-05
P10	**X6**	**6400**	-	-	138	-	-	7.121279	-	-	414	-	-	9.37E-05
	X7	**8100**	-	-	158	-	-	10.26578	-	-	474	-	-	9.60E-05
	X8	**12100**	-	-	160	-	-	13.64664	-	-	480	-	-	9.41E-05

Tab. 3: Winners with respect to iterations, function evaluations and CPU time

Method	SP	SDYP	PSG
TIME	7	14	48
ITER	0	22	50
FEVAL	0	22	50

Problem 4 [16]

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

Problem 5 Tridiagonal Exponential Problem [31]

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

Problem 6 Singular Function [32]

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

Problem 7 [30]

$$F(x) = \begin{pmatrix} 2 & -1 & \\ -1 & 2 & -1 \\ \ddots & \ddots & \ddots \\ \ddots & \ddots & -1 \\ & -1 & 2 \end{pmatrix} x + (e^{x_1} - 1, ..., e^{x_n} - 1)^T$$

Problem 8 [33]

$$F(x) = \begin{pmatrix} 5 & 3 & \\ 2 & 5 & 3 \\ \ddots & \ddots & \ddots \\ \ddots & \ddots & 3 \\ & 2 & 5 \end{pmatrix} x + (-1, -2, ..., -n)^T$$

Problem 9 [34] The function $F(x)$ is given by $F(x) = Ax + \phi(x)$, is the discretization of the boundary problem

$$\begin{cases} -\Delta u(x,y) = g(x,y,u), & (x,y) \in \Omega \subset \mathbb{R}^2 \\ u = 0, & (x,y) \in \partial\Omega \end{cases}$$

where Δ is the Laplace operator, $g(x,y,u) = -u^3 + 10$ and $\Omega = [0,1] \times [0,1]$. The matrix A is the bidimensional finite differences Laplacian $n \times n$ matrix given by

$$A = \begin{pmatrix} B & -I & \\ -I & B & -I \\ \ddots & \ddots & \ddots \\ \ddots & \ddots & -I \\ & -I & B \end{pmatrix} \in \mathbb{R}^{n \times n}, \tag{17}$$

where I is the $n_0 \times n_0$ identity matrix and

$$B = \begin{pmatrix} 4 & -1 & \\ -1 & 4 & -1 \\ \ddots & \ddots & \ddots \\ \ddots & \ddots & -1 \\ & -1 & 4 \end{pmatrix} \in \mathbb{R}^{n_0 \times n_0}.$$

$\phi(x) = h^2(x_1^3 - 10, ..., x_n^3 - 10)^T$, $n = n_0^2$ and $h = 1/(n_0 + 1)$.

Problem 10 [35] The function $F(x)$ is given by $F(x) = Ax + \psi(x) - b$, is the discretization of nondifferentiable Dirichlet problem which arise from magnetohydrodynamics equilibria. The matrix A is given by (17),

$$\psi(x) = -h^2(\max(\alpha x_1 - v, \beta x_1 - \mu), ..., \max(\alpha x_n - v, \beta x_n - \mu))^T,$$

where $h = 1/(n_0 + 1), n_0^2 = n, \alpha = v = 1, \beta = \mu = 0.5$ and $b = h^2(1, 1, ..., 1)^T$.

5 Conclusions

In this paper, we proposed a positive spectral gradient-like method for large-scale nonlinear monotone equations based upon a convex combination of two different spectral coefficients with projection technique, and it was proved to converge globally under some assumptions. Numerical results showed that the proposed algorithm is competitive to similar algorithms for large-scale problems. In addition, we have noticed that the choice of β affects the performance of the algorithms. Therefore, we suggest further research on the selection of β for an efficient algorithm.

Acknowledgements

We thank Professor Mehiddin Al-Baali for his careful reading of the manuscript and his comments which improved the paper. We are very grateful to the anonymous referee for his comments and suggestions, which we used to improve the earlier version of this paper.

References

[1] D.-H. Li, M. Fukushima, L. Qi, and N. Yamashita. Regularized newton methods for convex minimization problems with singular solutions. *Computational Optimization and Applications*, 28(2):131–147, 2004. DOI: 10.1023/B:COAP.0000026881.96694.32.

[2] N.A. Iusem and V.M. Solodov. Newton-type methods with generalized distances for constrained optimization. *Optimization*, 41(3):257–278, 1997. DOI: 10.1080/02331939708844339.

[3] F. Masao. Equivalent differentiable optimization problems and descent methods for asymmetric variational inequality problems. *Mathematical programming*, 53(1):99–110, 1992. DOI: 10.1007/BF01585696.

[4] Y.-B. Zhao and D. Li. Monotonicity of fixed point and normal mappings associated with variational inequality and its application. *SIAM Journal on Optimization*, 11(4):962–973, 2001. DOI: 10.1137/S1052623499357957.

[5] A.B. Abubakar and M.Y. Waziri. A matrix-free approach for solving systems of nonlinear equations. *Journal of Modern Methods in Numerical Mathematics*, 7(1):1–9, 2016. DOI: 10.20454/jmmnm.2016.1025.

[6] M. Al-Baali, E. Spedicato, and F. Maggioni. Broyden's quasi-newton methods for a nonlinear system of equations and unconstrained optimization: a review and open problems. *Optimization Methods and Software*, 29(5):937–954, 2014. DOI: 10.1080/10556788.2013.856909.

[7] C.G. Broyden. A class of methods for solving nonlinear simultaneous equations. *Math Comput*, 19:577–593, 1965. DOI: 10.1090/S0025-5718-1965-0198670-6.

[8] A. Cordero and J.R. Torregrosa. Variants of newton's method for functions of several variables. *Appl Math Comput*, 183(1):199–208, 2006. DOI: 10.1016/j.amc.2006.05.062.

[9] Ron S Dembo, Stanley C Eisenstat, and Trond Steihaug. Inexact newton methods. *SIAM Journal on Numerical analysis*, 19(2):400–408, 1982. DOI: 10.1137/0719025.

[10] H. Mohammad and M.Y. Waziri. On broyden-like update via some quadratures for solving nonlinear systems of equations. *Turkish Journal of Mathematics*, 39(3):335–345, 2015. DOI: 10.3906/mat-1404-41.

[11] N. Krejic and Z.. Luzanin. Newton-like method with modification of the right-hand-side vector. *Math Comput*, 71(237):237–250, 2002. DOI: 10.1090/S0025-5718-01-01322-9.

[12] M.Y. Waziri, W.J. Leong, and M.A. Hassan. Diagonal broyden-like method for large-scale systems of nonlinear equations. *Malaysian Journal of Mathematical Sciences*, 6(1):59–73, 2012. http://einspem.upm.edu.my/journal/fullpaper/vol6no1/4.

[13] M.Y. Waziri, W.J. Leong, M.A. Hassan, and M. Monsi. A new newton's method with diagonal jacobian approximation for systems of nonlinear equations. *Journal of Mathematics and Statistics*, 6(3):246–252, 2010. https://pdfs.semanticscholar.org/2e3e/a96f785191253c1fb9d60e333ceb3cb18c40.pdf.

[14] M.V. Solodov and B.F. Svaiter. A globally convergent inexact newton method for systems of monotone equations. In *Reformulation: Nonsmooth, Piecewise Smooth, Semismooth and Smoothing Methods*, pages 355–369. Springer, 1998. http://citeseerx.ist.psu.edu/viewdoc/download;jsessionid=6F4858DA35995C57CD1FBB62196FE07B?doi=10.1.1.47.7972&rep=rep1&type=pdf.

[15] G. Zhou and K.C. Toh. Superlinear convergence of a newton-type algorithm for monotone equations. *Journal of optimization theory and applications*, 125(1):205–221, 2005. DOI: 10.1007/s10957-004-1721-7.

[16] W. Zhou and D. Li. Limited memory bfgs method for nonlinear monotone equations. *Journal of Computational Mathematics*, 25(1):89–96, 2007. http://www.jstor.org/stable/43693347.

[17] L. Zhang and W. Zhou. Spectral gradient projection method for solving nonlinear monotone equations. *Journal of Computational and Applied Mathematics*, 196(2):478–484, 2006. DOI: 10.1016/j.cam.2005.10.002.

[18] J. Barzilai and J.M. Borwein. Two-point step size gradient methods. *IMA Journal of Numerical Analysis*, 8(1):141–148, 1988. DOI: 10.1093/imanum/8.1.141.

[19] W. La Cruz. A spectral algorithm for large-scale systems of nonlinear monotone equations. *Numerical Algorithms*, pages 1–22, 2017. DOI: 10.1007/s11075-017-0299-8.

[20] W. La Cruz, J. Martínez, and M. Raydan. Spectral residual method without gradient information for solving large-scale nonlinear systems of equations. *Mathematics of Computation*, 75:1429–1448, 2006. DOI: 10.1090/S0025-5718-06-01840-0.

[21] M. Ahookhosh, K. Amini, and S. Bahrami. Two derivative-free projection approaches for systems of large-scale nonlinear monotone equations. *Numerical Algorithms*, 64(1):21–42, 2013. DOI: 10.1007/s11075-012-9653-z.

[22] Z. Dai, X. Chen, and F. Wen. A modified perry's conjugate gradient method-based derivative-free method for solving large-scale nonlinear monotone equations. *Applied Mathematics and Computation*, 270(C):378–386, 2015. DOI: 10.1016/j.amc.2015.08.014.

[23] Q. Li and D.-H. Li. A class of derivative-free methods for large-scale nonlinear monotone equations. *IMA Journal of Numerical Analysis*, 31(4):1625–1635, 2011. DOI: 10.1093/imanum/drq015.

[24] J. Liu and S. Li. Spectral dy-type projection method for nonlinear monotone systems of equations. *Journal of Computational Mathematics*, 33(4):341–355, 2015. http://www.global-sci.org/jcm/galley/JCM4494.pdf.

[25] M. Li. A liu-storey-type method for solving large scale nonlinear monotone equations. *Numerical Functional Analysis and Optimization*, 35(3):310–322, 2014. DOI: 10.1080/01630563.2013.812656.

[26] Z. Papp and S. Rapajić. Fr type methods for systems of large-scale nonlinear monotone equations. *Applied Mathematics and Computation*, 269:816–823, 2015. DOI: 10.1016/j.amc.2015.08.002.

[27] Q.-R. Yan, X.-Z. Peng, and D.-H. Li. A globally convergent derivative-free method for solving large-scale nonlinear monotone equations. *Journal of computational and applied mathematics*, 234(3):649–657, 2010. DOI: 10.1016/j.cam.2010.01.001.

[28] W. Zhou and F. Wang. A PRP-based residual method for large-scale monotone nonlinear equations. *Applied Mathematics and Computation*, 261:1–7, 2015. DOI: 10.1016/j.amc.2015.03.069.

[29] Y.-H. Dai, M. Al-Baali, and X. Yang. A positive Barzilai-Borwein-like stepsize and an extension for symmetric linear systems. In *Numerical Analysis and Optimization*, pages 59–75. Springer, 2015. DOI: 10.1007/978-3-319-17689-5_3.

[30] W.-J. Zhou and D.-H. Li. A globally convergent bfgs method for nonlinear monotone equations without any merit functions. *Mathematics of Computation*, 77(264):2231–2240, 2008. http://www.jstor.org/stable/40234609.

[31] Yang Bing and Gao Lin. An efficient implementation of merrill's method for sparse or partially separable systems of nonlinear equations. *SIAM Journal on Optimization*, 1(2):206–221, 1991.

[32] W. La Cruz and M. Raydan. Nonmonotone spectral methods for large-scale nonlinear systems. *Optimization Methods and Software*, 18(5):583–599, 2003. DOI: 10.1080/10556780310001610493.

[33] W. Cheng. A prp type method for systems of monotone equations. *Mathematical and Computer Modelling*, 50(1-2):15–20, 2009. DOI: 10.1016/j.mcm.2009.04.007.

[34] J.M. Ortega and W.C. Rheinboldt. *Iterative Solution of Nonlinear Equations in Several Variables*. Academic Press, Inc., 1970. ISBN: 978-0-12-528550-6.

[35] J. Rappaz. Approximation of a nondifferentiable nonlinear problem related to mhd equilibria. *Numerische Mathematik*, 45(1):117–133, 1984. DOI: 10.1007/BF01379665.