International Mathematical Forum, Vol. 9, 2014, no. 36, 1767 - 1777
HIKARI Ltd, www.m-hikari.com
http://dx.doi.org/10.12988/imf.2014.411197

A Modified Liu-Storey Conjugate Gradient Projection Algorithm for Nonlinear Monotone Equations

Yaping Hu

School of Science, East China University of Science and Technology
Shanghai, 200237, China

Zengxin Wei

College of Mathematics and Information Science, Guangxi University
Nanning, 530004, China

Abstract

In this paper, a modified Liu-Storey (LS) conjugate gradient projection algorithm is proposed for solving nonlinear monotone equations based on a hyperplane projection technique. The proposed method is a derivative-free method and can be applied to solving large-scale nonsmooth equations for its lower storage requirement. We can establish its global convergence results under some suitable conditions. Numerical results show that this algorithm is efficient and promising.

Mathematics Subject Classification: 65K05, 90C26

Keywords: nonlinear monotone equations; conjugate gradient method; global convergence

1 Introduction

Considering the square nonlinear system of nonlinear monotone equations

$$g(x) = 0, \ x \in \Re^n, \tag{1}$$

where $g : \Re^n \to \Re^n$ is continuous and monotone, i.e. $\langle g(x)-g(y), x-y\rangle \geq 0$ for $\forall x, y \in \Re^n$. Nonlinear monotone equations have many practical background such as the first order necessary condition of the unconstrained convex optimization problem, the subproblems in the generalized proximal algorithms with Bregman distances [4], and economic equilibrium problems [2]. Different methods have been developed for nonlinear systems of monotone equations. Newton's method and quasi-Newton methods [9, 15] are particularly welcomed for their local superlinear convergence property. However, they are typically unattractive for large-scale nonlinear systems of equations because they need to solve a linear system by using the Jacobian matrix or an approximation of it. The method in [16] is based on the Barzilai-Borwein steplength. Li et al. [6] proposed derivative-free approaches based on modified PRP conjugate gradient techniques for solving (1). A prominent feature of these methods is that the search direction does not need gradient information, therefore these methods can be applied for nonsmooth equations. Recently, many numerical methods [3, 11, 13, 14] for nonlinear equations have been presented.

In this paper, we propose a modified Liu-Storey conjugate gradient projection algorithm for nonlinear monotone equations (1). This paper is organized as follows. In the next section, we discuss the modified LS conjugate gradient projection algorithm for nonlinear monotone equations and prove its global convergence. Some preliminary numerical results are presented in Section 3. Finally, we have a conclusion section.

2 Algorithm

First we recall the Liu-Storey(LS) conjugate gradient method [7] for unconstrained minimization problem:

$$\min_{x\in\Re^n} f(x), \tag{2}$$

where $f : \Re^n \to \Re$ is continuously differentiable. The iterative formula

$$x_{k+1} = x_k + \alpha_k d_k,\ k = 1, 2, \cdots \tag{3}$$

is often used in the CG method, where x_k is the kth iteration point, $\alpha_k > 0$ is the steplength, and d_k is the kth search direction defined by

$$d_{k+1} = \begin{cases} -g_{k+1} + \beta_k d_k, & \text{if } \ k \geq 1 \\ -g_{k+1}, & \text{if } \ k = 0, \end{cases} \tag{4}$$

where g_{k+1} is gradient of $f(x)$ at the point x_{k+1} and $\beta_k \in \Re$ is a scalar. The LS conjugate gradient formula is

$$d_{k+1} = \begin{cases} -g_{k+1} + \frac{g_{k+1}^T y_k}{-d_k^T g_k} d_k, & \text{if } \ k \geq 1 \\ -g_{k+1}, & \text{if } \ k = 0, \end{cases}$$

where $y_k = g_{k+1} - g_k$. Based on this method, we will present the following modified LS formula for nonlinear monotone equations (1)

$$d_{k+1} = \begin{cases} -g_{k+1} + \frac{g_{k+1}^T y_k d_k - d_k^T g_{k+1} y_k}{\max\{\gamma \|d_k\| \|y_k\|, d_k^T y_k, -d_k^T g_k\}}, & \text{if } k \geq 1 \\ -g_{k+1}, & \text{if } k = 0, \end{cases} \tag{5}$$

where $\gamma > 0$ is a constant. It is easy to see that the given algorithm can be reduced to a standard LS method if exact line search is used.

In general, an iterative method generates the next iteration by (3). For monotone equations, it is desirable to accelerate the iteration process by exploring the monotonicity of the equation. Let $z_k = x_k + \alpha_k d_k$. With the monotonicity of g, the hyperplane

$$P_k = \{x \in \Re^n \mid g(z_k)^T (x - z_k) = 0\}$$

strictly separates the current iteration x_k from the solution set of (1). Observing this fact, Solodov & Svaiter [9] advised letting the next iteration x_{k+1} be the projection of z_k onto this hyperplane. Specifically, x_{k+1} is determined by

$$x_{k+1} = x_k - \frac{g(z_k)^T (x_k - z_k)}{\|g(z_k)\|^2} g(z_k). \tag{6}$$

We will adopt this projection strategy to propose a modified LS conjugate gradient projection algorithm for solving nonlinear monotone equations (1). The steps of our model method are stated as follows.

Algorithm 2.1.
Step 0: Choose any initial point $x_0 \in \Re^n$, and constants $\rho \in (0,1)$, $\gamma > 0$, $\sigma > 0$, $s > 0$ and $\epsilon \in (0,1)$. Let $k := 0$.
Step 1: Stop if $\|g(x_k)\| \leq \epsilon$. Otherwise compute d_k by (5).
Step 2: Choose $\alpha_k = \max\{s\rho^i, i = 0, 1, \cdots\}$ such that

$$-g(x_k + s\rho^i d_k)^T d_k \geq \sigma s \rho^i \|g(x_k + \alpha_k d_k)\| \|d_k\|^2. \tag{7}$$

Step 3: Let $z_k = x_k + \alpha_k d_k$.
Step 4: If $\|g(z_k)\| \leq \epsilon$, stop and let $x_{k+1} = z_k$. Otherwise determine x_{k+1} by (6).
Step 5: Let $k := k + 1$, and go to Step 1.

Before we proceed to study some properties of Algorithm 2.1, we make the following assumption.
Assumption A (i) The solution set of the problem (1) is nonempty.
(ii) $g(x)$ is Lipschitz continuous on $\Re^n$, i.e. there exists a positive constant L such that

$$\|g(x) - g(y)\| \leq L\|x - y\|, \ \forall\, x,\, y \in \Re^n. \tag{8}$$

The following lemma shows that Algorithm 2.1 has the sufficiently descent property.

Lemma 2.1. *For all $k \geq 0$, we have*

$$g(x_k)^T d_k = -\|g(x_k)\|^2 \tag{9}$$

and

$$\|d_k\| \leq (1 + \frac{2}{\gamma})\|g(x_k)\|. \tag{10}$$

Proof. When $k = 0$, (9) and (10) hold since $d_0 = -g(x_0)$. From the definition of d_{k+1} in (5), we have

$$\begin{aligned} & d_{k+1}^T g(x_{k+1}) = -\|g(x_{k+1})\|^2 \\ & +[\frac{g(x_{k+1})^T y_k d_k - d_k^T g(x_{k+1}) y_k}{\max\{\gamma\|d_k\|\|y_k\|, d_k^T y_k, -d_k^T g(x_k)\}}]^T g(x_{k+1}) \\ & = -\|g(x_{k+1})\|^2. \end{aligned}$$

Thus (9) holds for all $k \geq 1$, and

$$\begin{aligned} & \|d_{k+1}\| = \| - g(x_{k+1}) + \frac{g(x_{k+1})^T y_k d_k - d_k^T g(x_{k+1}) y_k}{\max\{\gamma\|d_k\|\|y_k\|, d_k^T y_k, -d_k^T g(x_k)\}}\| \\ & \leq \|g(x_{k+1})\| + \frac{\|g(x_{k+1})\|\|y_k\|\|d_k\| + \|d_k\|\|g(x_{k+1})\|\|y_k\|}{\max\{\gamma\|d_k\|\|y_k\|, d_k^T y_k, -d_k^T g(x_k)\}} \\ & \leq (1 + \frac{2}{\gamma})\|g(x_{k+1})\|, \end{aligned}$$

where the last inequality follows from

$$\max\{\gamma\|d_k\|\|y_k\|, d_k^T y_k, -d_k^T g(x_k)\} \geq \gamma\|d_k\|\|y_k\|.$$

Then (10) holds. □

Lemma 2.2. *Let Assumption A holds. Then Algorithm 2.1 will produce an iteration $z_k = x_k + \alpha_k d_k$ in a finite number of backtracking steps.*

Proof. Suppose that $\|g(x_k)\| \to 0$ does not hold, or Algorithm 2.1 stops. Then there exists a constant $\epsilon_0 > 0$ satisfying

$$\|g(x_k)\| \geq \epsilon_0, \forall\, k \geq 0. \tag{11}$$

We prove this lemma by contradiction. Suppose that the condition (7) does not hold for some iteration indexes k_*. Let $\alpha_{k_*}^{(m)} = \rho^m s$, it can be concluded

$$-g(x_{k_*} + \alpha_{k_*}^{(m)} d_{k_*})^T d_{k_*} < \sigma\alpha_{k_*}^{(m)}\|g(x_{k_*} + \alpha_{k_*}^{(m)} d_{k_*})\|\|d_{k_*}\|^2, \forall\, m \geq 0.$$

By Assumption A and (9) in Lemma 2.1, we have

$$\begin{aligned} \|g(x_k)\|^2 &= -g(x_k)^T d_k \\ &= [g(x_{k_*} + \alpha_{k_*}^{(m)} d_{k_*}) - g(x_k)]^T d_k - g(x_{k_*} + \alpha_{k_*}^{(m)} d_{k_*})^T d_{k_*} \\ &< [L + \sigma \|g(x_{k_*} + \alpha_{k_*}^{(m)} d_{k_*})\|] \alpha_{k_*}^{(m)} \|d_{k_*}\|^2, \forall\, m \geq 0. \end{aligned} \tag{12}$$

By Assumption A (ii), we conclude that there exists a constant $M > 0$ satisfying

$$\|g(x_k)\| \leq M. \tag{13}$$

Then it follows from (10) that

$$\begin{aligned} \|g(x_{k_*} + \alpha_{k_*}^{(m)} d_{k_*})\| &\leq \|g(x_{k_*} + \alpha_{k_*}^{(m)} d_{k_*}) - g_k\| + \|g_k\| \\ &\leq L\alpha_{k_*}^{(m)} \|d_{k_*}\| + M \\ &\leq LMs(1 + \frac{2}{\gamma}) + M. \end{aligned}$$

Thus, for $\forall\, m \geq 0$, we have

$$\alpha_{k_*}^{(m)} > \frac{\|g_k\|^2}{[L + \sigma \|g(x_{k_*} + \alpha_{k_*}^{(m)} d_{k_*})\|] \|d_{k_*}\|^2} > \frac{\epsilon_0^2}{[L + LMs(1 + \frac{2}{\gamma}) + M](M + \frac{2M}{\gamma})^2} > 0,$$

which contradicts with the definition of $\alpha_{k_*}^{(m)}$. Consequently, the line search (7) can attain a positive steplength α_k in a finite number of backtracking repetitions. □

The above lemma shows that the line search of Algorithm 2.1 is reasonable, namely, the given MLS conjugate gradient projection algorithm is well defined. Similar to Lemma 2.3 in [12] and Lemma 2.1 in [9], it is not difficult to get the following lemma, so we omit the proof.

Lemma 2.3. *Let Assumption A holds and the sequence $\{x_k, z_k\}$ be generated by Algorithm 2.1. Suppose that x^* is a solution of problem (1) with $g(x^*) = 0$. Then*

$$\|x_{k+1} - x^*\| \leq \|x_k - x^*\|^2 - \sigma^2 \|x_k - z_k\|^4. \tag{14}$$

In particular, both $\{x_k\}$ and $\{z_k\}$ are bounded. Furthermore,

$$\lim_{k \to \infty} \|x_k - z_k\| = 0, \tag{15}$$

$$\lim_{k \to \infty} \|x_{k+1} - x_k\| = 0, \tag{16}$$

Theorem 2.4. *Let Assumption A holds and the sequence $\{x_k\}$ be generated by Algorithm 2.1. Then we have*

$$\lim_{k \to \infty} \inf \|g_k\| = 0. \tag{17}$$

Proof. Suppose that (17) is not true. Let $\epsilon_0 > 0$ be a constant such that $\|g_k\| \geq \epsilon_0$. This together with (9) implies that

$$\|d_k\| \geq \|g_k\| \geq \epsilon_0, \forall\ k \geq 0. \tag{18}$$

By the relation of (5), (10) and (13), we obtain

$$\|d_{k+1}\| \leq (1 + \frac{2}{\gamma})\|g_{k+1}\| \leq (1 + \frac{2}{\gamma})M, \forall\ k \geq 0, \tag{19}$$

which implies that the sequence $\{\|d_k\|\}$ is bounded. Then there exists an infinite index set N_1 and an accumulation point $\bar{d}$ satisfying

$$\lim_{k\to\infty} d_k = \bar{d}, \ \text{ for } k \in N_1.$$

By the boundedness of $\{x_k\}$ in Lemma 2.3, we can deduce that there exists an accumulation $\bar{x}$ and an infinite index set $N_2 \subset N_1$ satisfying

$$\lim_{k\to\infty} x_k = \bar{x}, \ \text{ for } k \in N_2.$$

By Lemma 2.2 and Lemma 2.3, we get

$$\alpha_k\|d_k\| \to 0, \ k \to \infty,$$

this together with (19), we obtain $\lim_{k\to\infty} \alpha_k = 0$.
By (7), we have

$$-g(x_k + \alpha'_k d_k)^T d_k < \sigma\alpha'_k\|g(x_k + \alpha'_k d_k)\|\|d_k\|^2, \tag{20}$$

where $\alpha'_k = \frac{\alpha_k}{\rho}$. Therefore, taking the limit as $k \to \infty$ in both sides of (20) for all $k \in N_2$ generates

$$g(\bar{x})^T\bar{d} > 0.$$

On the other hand, by taking the limit as $k \to \infty$ in both sides of (9) for all $k \in N_2$, we have

$$g(\bar{x})^T\bar{d} \leq 0,$$

which generates a contradiction. The proof is complete. □

3 Numerical Experiments

In this section, we do some numerical experiments to test the performance of the MLS conjugate gradient projection method and compare it with the modified Polak-Ribière-Polyak (MPRP) projection method in [6] and the spectral gradient (SG) projection method in [16]. All of the algorithms are coded in

MATLAB R2010a and run on a personal computer with Intel Core 2 Duo CPU at 2.8 GHz and 2 GB of memory. The test problems are listed as $g(x) = (g_1(x), g_2(x), \cdots, g_n(x))^T$, which have the associated initial guess x_0.

Problem 1. The function is taken from [5].

$$\begin{aligned} g_1(x) &= 2.5x_1 + x_2 - 1, \\ g_i(x) &= x_{i-1} + 2.5x_i + x_{i+1} - 1, \ i = 2, 3, \cdots, n-1, \\ g_n(x) &= x_{n-1} + 2.5x_n - 1. \end{aligned}$$

Initial guess: $x_0 = (3, 3, \cdots, 3)^T$.

Problem 2. (Wang et al. [10]). $g(x) : R^n \to R^n$ with

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

Initial guess: $x_0 = (\frac{1}{n}, \frac{1}{n}, \cdots, \frac{1}{n})^T$.

Problem 3. (Zhang & Zhou, [16]). $g : R^n \to R^n$ is given by

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

Initial guess: $x_0 = (1, 1, \cdots, 1)^T$.

Problem 4. Discrete boundary value problem [8].

$$\begin{aligned} g_1(x) &= 2x_1 + 0.5h^2(x_1 + h)^3 - x_2, \\ g_i(x) &= 2x_i + 0.5h^2(x_i + hi)^3 - x_{i-1} + x_{i+1}, \\ & \quad i = 2, 3, \cdots, n-1 \\ g_n(x) &= 2x_n + 0.5h^2(x_n + hn)^3 - x_{n-1}, \\ h &= \frac{1}{n+1}. \end{aligned}$$

Initial guess: $x_0 = (1, 1, \cdots, 1)$.

Problem 5. The example is taken from [12].

$$\begin{aligned} g_1(x) &= x_1 - e^{cos(\frac{x_1+x_2}{n+1})}, \\ g_i(x) &= x_i - e^{cos(\frac{x_{i-1}+x_i+x_{i+1}}{n+1})}, \ i = 2, 3, \cdots, n-1, \\ g_n(x) &= x_n - e^{cos(\frac{x_{n-1}+x_n}{n+1})}. \end{aligned}$$

Initial guess: $x_0 = (0.1, 0.1, \cdots, 0.1)$.

For MLS projection method, the parameters are specified as $\sigma = 0.01$, $s = 1$, $\rho = 0.6$, $\gamma = 0.5$, and $\epsilon = 10^{-4}$. For MPRP projection method in [6] and SG projection method in [16], we use the default parameters there. For all methods, we adopt the termination condition $\|g(x_k)\| \leq \epsilon$. The columns of Tables 3.1 and 3.2 have the following meanings:

Dim: the dimension of the problem. NI: the total number of iterations.

NG: the number of the function evaluations. Time: CPU time in seconds.

GF: the final function norm evaluations when the program is terminated.

We compared these three methods on problems 1-5. The results are reported in Tables 3.1 and 3.2. Table 3.1 shows the results of the methods with given initial points for problems 1-5. Table 3.2 lists the results obtained by the methods with random initial points generated by Matlab's code $rand(n, 1)$. We see from Tables 3.1 and 3.2 that, all of these three methods terminate successfully at a solution of the problem starting from any initial point. Moreover, Tables 3.1 and 3.2 also indicate that the dimensions of these problems do not affect the number of iterations of the algorithm. However, for high dimension case, the computing time is relatively large. Taking everything together, the numerical experiments show that the proposed method performs well for solving the given systems of monotone equations.

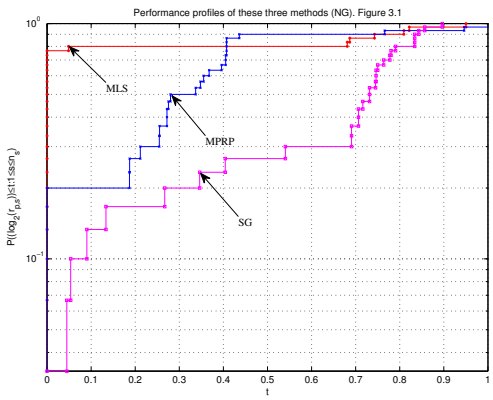
TABLE 3.1. Numerical Results for Problems 1-5 with given initial points

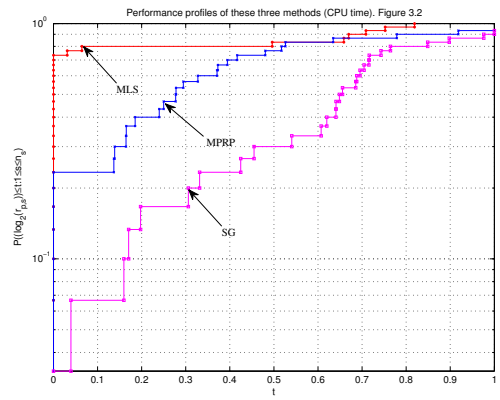
		MLS		MPRP		SG	
P	Dim	NI/NG/Time	GF	NI/NG/Time	GF	NI/NG/Time	GF
1	1000	43/238/0.749	6.961621e-005	61/271/1.045	9.867416e-005	55/247/0.843	9.661741e-005
	5000	44/241/10.936	7.828087e-005	63/282/12.168	7.443931e-005	50/233/10.702	8.864162e-005
	10000	46/253/38.282	8.042732e-005	64/288/51.105	7.172019e-005	56/261/48.173	8.307898e-005
2	1000	7/22/0.109	5.174617e-005	9/28/0.156	6.170168e-005	12/37/0.218	6.868747e-005
	5000	6/19/1.077	5.791179e-005	8/25/1.03	5.523173e-005	10/31/1.919	8.547541e-005
	10000	6/19/3.369	4.095491e-005	7/22/3.776	7.811728e-005	10/31/6.068	6.045321e-005
3	1000	14/43/0.203	6.293830e-005	18/55/0.249	9.763268e-005	24/73/0.312	9.455022e-005
	5000	15/46/2.481	5.629373e-005	20/61/3.573	5.457833e-005	26/79/3.868	7.611146e-005
	10000	15/46/9.937	7.961136e-005	20/61/12.043	7.718541e-005	27/82/16.318	6.458271e-005
4	1000	38/169/0.764	8.694308e-005	25/101/0.484	8.664148e-005	47/188/0.952	6.150068e-005
	5000	38/169/9.22	8.487935e-005	26/105/6.537	7.590343e-005	44/176/8.081	8.826866e-005
	10000	42/184/34.882	7.121343e-005	26/105/21.341	8.657642e-005	45/180/33.618	9.194040e-005
5	1000	15/46/0.297	8.878038e-005	20/61/0.327	7.886463e-005	27/82/0.484	8.455840e-005
	5000	17/53/2.871	5.088791e-005	21/64/3.603	8.827727e-005	29/88/4.649	6.820980e-005
	10000	18/57/12.153	4.606058e-005	22/68/14.352	9.363581e-005	30/92/19.547	8.103480e-005

TABLE 3.2. Numerical Results for Problems 1-5 with random initial points

		MLS		MPRP		SG	
P	Dim	NI/NG/Time	GF	NI/NG/Time	GF	NI/NG/Time	GF
1	1000	33/182/0.717	7.722104e-005	87/374/1.825	7.301800e-005	52/219/1.124	9.908726e-005
	5000	44/237/10.639	6.547100e-005	94/403/20.109	7.610513e-005	62/260/12.2	9.070390e-005
	10000	43/232/45.989	8.738841e-005	105/447/92.665	8.427473e-005	61/247/47.268	9.879317e-005
2	1000	9/28/0.109	4.823992e-005	11/34/0.187	8.998248e-005	15/46/0.234	8.588589e-005
	5000	10/31/2.028	4.271021e-005	12/37/2.23	9.952395e-005	17/52/3.089	6.840170e-005
	10000	10/31/6.646	5.963495e-005	13/40/8.736	6.947915e-005	17/52/11.123	9.550824e-005
3	1000	11/34/0.156	7.780008e-005	15/46/0.202	5.659040e-005	20/61/0.265	6.767709e-005
	5000	12/37/2.574	6.889901e-005	16/49/2.886	6.258755e-005	21/64/3.744	8.982259e-005
	10000	12/37/6.693	9.619871e-005	16/49/10.39	8.738470e-005	22/67/13.65	7.524707e-005
4	1000	44/194/0.999	9.955959e-005	30/121/0.593	9.158351e-005	44/176/0.796	8.024448e-005
	5000	53/228/11.513	7.181482e-005	32/129/7.238	8.282484e-005	41/164/9.922	9.237144e-005
	10000	60/257/50.996	6.145535e-005	33/133/28.907	7.601912e-005	44/176/32.292	9.995324e-005
5	1000	15/46/0.266	7.610934e-005	20/61/0.344	6.762036e-005	27/82/0.437	7.240755e-005
	5000	17/53/4.29	5.273351e-005	21/64/5.102	9.147879e-005	29/88/6.911	7.068359e-005
	10000	18/57/17.098	4.773259e-005	23/72/20.732	7.277609e-005	30/92/26.661	8.397638e-005

In order to analyze the efficiency of these three methods, we also use the tool of Dolan and Moré [1]. Figures 3.1 and 3.2 show the performance of these three methods relative to NG and CPU time of Tables 3.1-3.2, respectively. These two figures show that all of these three methods have good performance for all problems. And the proposed method is more competitive than the MPRP and SG projection methods as MLS projection method can get the solution of all the test problem at a smaller horizontal axis.





4 CONCLUSION

In this paper, we develop a modified LS conjugate gradient projection method for solving nonlinear monotone equations. The proposed method is particularly suitable for large-scale monotone equations because of low memory requirement. The global convergence of the given algorithm is established under suitable conditions. The preliminary numerical results show that our method is promising.

Acknowledgements. This work is supported by China NSF (Grant No. 11161003 11261006) and Guangxi NSF (Grant No. 2012GXNSFAA053002).

References

[1] E. Dolan, and J. Moré, Benchmarking optimization software with performance profiles, Math. Program., 91(2002), 201-213. http://dx.doi.org/10.1007/s101070100263

[2] S.P. Dirkse, and M.C. Ferris, MCPLIB: A collection of nonlinear mixed complementarity problems, Optim. Meth. Soft., 5(1995), 319-345. http://dx.doi.org/10.1080/10556789508805619

[3] Y.P. Hu, and Z.X. Wei, Wei-Yao-Liu conjugate gradient projection algorithm for nonlinear monotone equations with convex constraints, International Journal of Computer Mathematics, 2014, http:// dx.doi.org/ 10.1080/00207160.2014.977879

[4] A.N. Iusem, and M.V. Solodov, Newton-type methods with generalized distances for constrained optimization, Optim., 41(1997), 257-278. http://dx.doi.org/10.1080/02331939708844339

[5] M. Li, An Liu-Storey-Type Method for Solving Large-Scale Nonlinear Monotone Equations, Numerical Functional Analysis and Optimization, 35(2014), 310-322. http://dx.doi.org/10.1080/01630563.2013.812656

[6] Q.N. Li, and D.H. Li, A class of derivative-free methods for large-scale nonlinear monotone equations, IMA Journal of Numerical Analysis, 31(2011), 1625-1635. http://dx.doi.org/10.1093/imanum/drq015

[7] Y. Liu, and C. Storey, Efficient generalized conjugate gradient algorithms, Part 1: Theory, J. Optim. Theory Appl., 69(1991), 129-137. http://dx.doi.org/10.1007/bf00940464

[8] J. Moré, B. Garbow, and K. Hillström, Testing unconstrained optimization software, ACM T. Math. Software, 7(1981), 17-41. http://dx.doi.org/10.1145/355934.355936

[9] M. V. Solodov, and B. F. Svaiter, A globally convergent inexact Newton method for systems of monotone equations. in: M. Fukushima, L. Qi (Eds.), Reformulation: Nonsmooth, Piecewise Smooth, Semismooth and Smoothing Methods, Kluwer Academic Publishers, (1998), 355-369. http://dx.doi.org/10.1007/978-1-4757-6388-1_18

[10] C. W. Wang, Y. J. Wang, and C. L. Xu, A projection method for a system of nonlinear monotone equations with convex constraints, Math. Meth. Oper. Res., 66(2007), 33-46. http://dx.doi.org/10.1007/s00186-006-0140-y

[11] Y. H. Xiao, and H. Zhu, A conjugate gradient method to solve convex constrained monotone equations with applications in compressive sensing, Journal of Mathematical Analysis and Applications, 405(2013), 310-319. http://dx.doi.org/10.1016/j.jmaa.2013.04.017

[12] G. H. Yu, S. Z. Niu, and J. H. Ma, Multivariate spectral gradient projection method for nonlinear monotone equations with convex constraints, Journal of Industrial and Management Optimization, 9(2013), 117-129. http://dx.doi.org/10.3934/jimo.2013.9.117

[13] G. H. Yu, Nonmonotone Spectral Gradient-type Methods for Large-scale Unconstrained Optimization and Nonlinear Systems of Equations, Pacific Journal of Optimization, 7(2011), 387-404.

[14] G. L. Yuan, Z. X. Wei, and X. W. Lu, A BFGS trust-region method for nonlinear equations, Computing, 92((2011), 317-333. http://dx.doi.org/10.1007/s00607-011-0146-z

[15] M. Zhang, Y. H. Xiao, and H. Dou, Solving nonlinear constrained monotone equations via limited memory BFGS algorithm, Journal of Computational Information Systems, 7(2011), 3995-4006.

[16] L. Zhang, and W. J. Zhou, Spectral gradient projection method for solving nonlinear monotone equations, Journal of Computational and Applied Mathematics, 196(2006), 478-484. http://dx.doi.org/10.1016/j.cam.2005.10.002

Received: December 11, 2014; Published: December 30, 2014