Wu *Journal of Inequalities and Applications* 2014, 2014:248
http://www.journalofinequalitiesandapplications.com/content/2014/1/248

Journal of Inequalities and Applications
a SpringerOpen Journal

RESEARCH Open Access

Global convergence of a modified conjugate gradient method

Xuesha Wu[*]

[*]Correspondence:
wuxuesha2013@126.com
College of General Education, Chongqing College of Electronic Engineering, Chongqing, 401331, P.R. China

Abstract

A modified conjugate gradient method to solve unconstrained optimization problems is proposed which satisfies the sufficient descent condition in the case of the strong Wolfe line search, and its global convergence property is established simply. The numerical results show that the proposed method is promising for the given test problems.

MSC: 90C26; 65H10

Keywords: unconstrained optimization; conjugate gradient method; sufficient descent condition; global convergence

1 Introduction

The nonlinear conjugate gradient method is one of the best methods to solve unconstrained optimization problems. It comprises a class of unconstrained optimization algorithms which is characterized by low memory requirements and strong local or global convergence properties. Therefore, a modified nonlinear conjugate gradient method is proposed and analyzed in this paper.

Consider the following unconstrained optimization problem:

$$\min_{x \in R^n} f(x), \tag{1.1}$$

where $f: R^n \to R$ is a smooth function and its gradient is denoted by g.

The conjugate gradient methods for solving the above problem often use the following iterative rules:

$$x_{k+1} = x_k + \alpha_k d_k, \tag{1.2}$$

where x_k is the current iterate, the stepsize α_k is a positive scalar which is generated by some line search, and the search direction d_k is defined by

$$d_k = \begin{cases} -g_k, & \text{for } k = 1; \\ -g_k + \beta_k d_{k-1}, & \text{for } k \geq 2, \end{cases} \tag{1.3}$$

where $g_k = \nabla f(x_k)$, β_k is the conjugate parameter which determines the performances of the corresponding methods. There are many well-known parameters β_k, such as

$$\beta_k^{\mathrm{PRP}} = \frac{g_k^T(g_k - g_{k-1})}{\|g_{k-1}\|^2} \quad \big(\text{Polak-Ribière-Polyak (PRP) [1, 2]}\big),$$

$$\beta_k^{\mathrm{LS}} = -\frac{g_k^T(g_k - g_{k-1})}{d_{k-1}^T g_{k-1}} \quad \text{(Liu-Storey (LS) [3])},$$

$$\beta_k^{\mathrm{HZ}} = \left(y_{k-1} - 2d_{k-1}\frac{\|y_{k-1}\|^2}{d_{k-1}^T y_{k-1}}\right)^T \frac{g_k}{d_{k-1}^T y_{k-1}} \quad \text{(Hager-Zhang [4])},$$

where $\|\cdot\|$ is the Euclidean norm. Their corresponding methods are generally called PRP, LS, and HZ conjugate gradient methods. If f is a strictly convex quadratic function, these methods are equivalent in the case that an exact line search is used. If f is non-convex, their behaviors may show some differences.

When the objective function is convex, Polak and Ribière [1] proved that the PRP method is globally convergent under the exact line search. But Powell [5] showed that the PRP method does not converge globally for some non-convex functions. However, in the past few years, the PRP method is generally believed to be the most efficient conjugate gradient method in practical computation. One remarkable property of the PRP method is that it essentially performs a restart if a bad direction occurs (see [6]). But Powell [5] constructed an example showing that the PRP method can cycle infinitely without approaching any stationary point even if an exact line search is used. This counter-example also indicates that the PRP method has a drawback in that it may not globally be convergent when the objective function is non-convex. Recently, Zhang *et al.* [7] proposed a descent modified PRP conjugate gradient method and proved its global convergence. The LS method has a similar property as the PRP method. The global convergence of the LS method with the Grippo-Lucidi line search has also been proved in [8]. Some researchers have further studied the LS method (see Liu [9], Liu and Du [10]). In addition, Hager and Zhang [4] gave another effective method, namely the CG-DESCENT method. It not only has stable convergence, but it also shows an effective numerical experiment result. In this method, the parameter β_k is computed by $\beta_k = \max\{\beta_k^{\mathrm{HZ}}, \eta_k\}$, where $\eta_k = \frac{-1}{\|d_{k-1}\| \min\{\eta, \|g_{k-1}\|\}}$, $\eta > 0$.

In the next section, a modified conjugate gradient method is proposed. In Section 3, we prove the global convergence of the proposed method for non-convex functions in the case of the strong Wolfe line search. In Section 4, we report some numerical results.

2 The new algorithm

Recently, some people have studied some variants of the LS method. For example, Li *et al.* [11] proposed a modified LS method where the parameter β_k is computed by

$$\beta_k = -\frac{g_k^T(g_k - g_{k-1})}{d_{k-1}^T g_{k-1}} - t\frac{\|g_k - g_{k-1}\|^2 d_{k-1}^T g_k}{(d_{k-1}^T g_{k-1})^2},$$

where $t > \frac{1}{4}$ is a constant. They proved the global convergence of the modified method with the Armijo line search and Wolfe line search. Tang *et al.* [12] proved the LS method with the new line search. Liu *et al.* [13] studied a modified LS method where the parameter β_k is computed by

$$\beta_k^{\mathrm{LS2}} = \begin{cases} \frac{g_k^T(g_k - g_{k-1})}{\rho|g_k^T d_{k-1}| - g_{k-1}^T d_{k-1}} & \text{if } \min\{1, \rho - 1 - \xi\} \cdot \|g_k\|^2 > |g_k^T g_{k-1}|, \\ 0 & \text{else}, \end{cases}$$

where $\rho > 1 + \xi$, $\xi > 0$. They proved the global convergence of the corresponding method with the Wolfe line search. In 2006, Wei *et al.* [14] proposed a modified PRP method where the parameter β_k is obtained by

$$\beta_k = \frac{g_k^T(g_k - \frac{\|g_k\|}{\|g_{k-1}\|}g_{k-1})}{\|g_{k-1}\|^2}.$$

They proved its global convergence with the exact line search, the strong Wolfe line search, and the Grippo-Lucidi line search, respectively. Their work overcomes the weak convergence of the PRP method. Inspired by their work, we consider a variant of LS method, *i.e.*

$$\beta_k^{\mathrm{VLS}} = \frac{g_k^T(g_k - t_k g_{k-1})}{-\lambda d_{k-1}^T g_{k-1} + (1-\lambda)\max\{0, g_k^T d_{k-1}\}}, \tag{2.1}$$

where $t_k = \frac{\|g_k\|}{\|g_{k-1}\|}$, $\lambda \in (0,1)$ and $\lambda > 2\sigma$. Obviously, the denominator of (2.1) is a convex combination of $-d_{k-1}^T g_{k-1}$ and $\max\{0, g_k^T d_{k-1}\}$ which may avoid the denominator of β_k^{LS} tending to zero. Now, we state formally the corresponding algorithm scheme for unconstrained optimization problems.

Algorithm 2.1

Step 0: Given an initial $x_1 \in R^n$, $\varepsilon \geq 0$, $\lambda = 0.8$. Set $k = 1$.

Step 1: If $\|g_1\| \leq \varepsilon$, then stop.

Step 2: Compute α_k by the strong Wolfe line search ($0 < \delta < \sigma < \frac{1}{2}$):

$$f(x_k + \alpha_k d_k) \leq f(x_k) + \delta \alpha_k g_k^T d_k, \tag{2.2}$$

$$\left|g(x_k + \alpha_k d_k)^T d_k\right| \leq -\sigma g_k^T d_k. \tag{2.3}$$

Step 3: Let $x_{k+1} = x_k + \alpha_k d_k$, $g_{k+1} = g(x_{k+1})$, if $\|g_{k+1}\| \leq \varepsilon$, then stop.

Step 4: Compute β_{k+1} by (2.1), and generate d_{k+1} by (1.3).

Step 5: Set $k = k + 1$, go to step 2.

In some references, the sufficient descent condition

$$g_k^T d_k \leq -c\|g_k\|^2, \quad c > 0, \tag{2.4}$$

is always assumed to hold. Because it plays an important role in proving the global convergence of conjugate gradient methods. Fortunately, in this paper, the search direction d_k satisfies the sufficient descent condition in the case of the strong Wolfe line search without any assumption.

Lemma 2.1 *Let the sequences* $\{g_k\}$ *and* $\{d_k\}$ *be generated by Algorithm* 2.1, *then we obtain*

$$g_k^T d_k \leq -\left(1 - \frac{2\sigma}{\lambda}\right) \cdot \|g_k\|^2. \tag{2.5}$$

Proof The conclusion can be proved by induction. Since $g_1^T d_1 = -\|g_1\|^2$, the conclusion (2.5) holds for $k = 1$. Now we assume that the conclusion (2.5) holds for $k \geq 1$ and $g_{k+1} \neq 0$.

One gets from (1.3) that

$$\begin{aligned}
-\frac{g_{k+1}^T d_{k+1}}{\|g_{k+1}\|^2} &= 1-\beta_{k+1}^{\mathrm{VLS}}\frac{g_{k+1}^T d_k}{\|g_{k+1}\|^2} \geq 1-\left|\beta_{k+1}^{\mathrm{VLS}}\right|\cdot\frac{|g_{k+1}^T d_k|}{\|g_{k+1}\|^2} \\
&\geq 1-\frac{\|g_{k+1}\|^2+\frac{\|g_{k+1}\|}{\|g_k\|}\cdot|g_{k+1}^T g_k|}{\lambda|g_k^T d_k|}\cdot\frac{|g_{k+1}^T d_k|}{\|g_{k+1}\|^2} \\
&\geq 1-\frac{2\|g_{k+1}\|^2}{\lambda|g_k^T d_k|}\cdot\frac{\sigma|g_k^T d_k|}{\|g_{k+1}\|^2} = 1-\frac{2\sigma}{\lambda}.
\end{aligned}$$

From the above inequality, the conclusion (2.5) holds for $k+1$. Thus, the conclusion (2.5) holds for $k \in N^+$. □

Remark 2.1 From (2.5) and the definition of β_k^{VLS}, it is not difficult to find that

$$\begin{aligned}
\beta_k^{\mathrm{VLS}} &= \frac{g_k^T(g_k-t_k g_{k-1})}{-\lambda d_{k-1}^T g_{k-1}+(1-\lambda)\max\{0, g_k^T d_{k-1}\}} \\
&\geq \frac{\|g_k\|^2-\frac{\|g_k\|}{\|g_{k-1}\|}\cdot|g_k^T g_{k-1}|}{-\lambda d_{k-1}^T g_{k-1}+(1-\lambda)\max\{0, g_k^T d_{k-1}\}} \\
&\geq \frac{\|g_k\|^2-\frac{\|g_k\|}{\|g_{k-1}\|}\cdot\|g_k^T\|\cdot\|g_{k-1}\|}{-\lambda d_{k-1}^T g_{k-1}+(1-\lambda)\max\{0, g_k^T d_{k-1}\}} = 0.
\end{aligned}$$

3 Global convergence of Algorithm 2.1

In order to prove the global convergence of Algorithm 2.1, the following assumptions for the objective function are often used.

Assumption (H)

(i) The level set $\Omega = \{x \mid f(x) \leq f(x_1)\}$ is bounded, where x_1 is the starting point.

(ii) In some neighborhood V of Ω, the objective function f is continuously differentiable, and its gradient is Lipschitz continuous, *i.e.*, there exists a constant $L > 0$ such that

$$\left\|g(x)-g(y)\right\| \leq L\|x-y\|, \quad \text{for all } x, y \in V. \tag{3.1}$$

From Assumption (H), there exists a constant $\tilde{r} > 0$ such that

$$\|g_k\| \leq \tilde{r}, \quad \text{for all } k.$$

The conclusion of the following lemma, often called the Zoutendijk condition, is usually used to prove the global convergence properties of conjugate gradient methods. It was originally established by Zoutendijk [15].

Lemma 3.1 *Suppose Assumption* (H) *holds. Let the sequences* $\{g_k\}$ *and* $\{d_k\}$ *be generated by Algorithm* 2.1, *then we have*

$$\sum_{k\geq 1}\frac{(g_k^T d_k)^2}{\|d_k\|^2} < +\infty. \tag{3.2}$$

Lemma 3.2 *Suppose Assumption* (H) *holds. Let the sequences* $\{g_k\}$ *and* $\{d_k\}$ *be generated by Algorithm* 2.1, *and let there exist a constant* $r > 0$ *such that*

$$\|g_k\| \geq r, \quad \text{for all } k \geq 1. \tag{3.3}$$

Then we have

$$\sum_{k\geq 2} \|u_k - u_{k-1}\|^2 < +\infty, \quad u_k = \frac{d_k}{\|d_k\|}.$$

Proof This lemma can be proved in a similar way as in [16], so we omit it. □

Lemma 3.3 *Suppose Assumption* (H) *holds. Let the sequences* $\{g_k\}$ *and* $\{d_k\}$ *be generated by Algorithm* 2.1, *and let the sequence* $\{g_k\}$ *satisfy*

$$0 < r \leq \|g_k\| \leq \tilde{r}, \quad \text{for all } k \geq 1. \tag{3.4}$$

Then the conjugate parameter β_k^{VLS} *has property* $(*)$, *i.e.*,

(1) *there exists a constant* $b > 1$ *such that* $|\beta_k^{\mathrm{VLS}}| \leq b$;
(2) *there exists a constant* $\tau > 0$, *such that* $\|x_k - x_{k-1}\| \leq \tau \Rightarrow |\beta_k^{\mathrm{VLS}}| \leq \frac{1}{2b}$.

Proof It follows from (2.1), (3.4), and (2.5) that

$$\begin{aligned}
|\beta_k^{\mathrm{VLS}}| &= \left|\frac{g_k^T(g_k - t_k g_{k-1})}{-\lambda d_{k-1}^T g_{k-1} + (1-\lambda)\max\{0, g_k^T d_{k-1}\}}\right| \\
&\leq \frac{\|g_k\|(\|g_k\| + \frac{\tilde{r}}{r}\cdot\|g_{k-1}\|)}{\lambda|d_{k-1}^T g_{k-1}|} \\
&\leq \frac{\|g_k\|(\|g_k\| + \frac{\tilde{r}}{r}\cdot\|g_{k-1}\|)}{(\lambda - 2\sigma)\|g_{k-1}\|^2} \leq \frac{\tilde{r}(\tilde{r} + \frac{\tilde{r}^2}{r})}{(\lambda - 2\sigma)r^2} \\
&\leq \frac{\tilde{r}^2(r + \tilde{r})}{(\lambda - 2\sigma)r^3} = b.
\end{aligned}$$

Define $\tau = \frac{(\lambda-2\sigma)r^2}{4L\tilde{r}b}$. Let $\|x_k - x_{k-1}\| \leq \tau$, it then follows from Assumption (H)(ii) that

$$\begin{aligned}
|\beta_k^{\mathrm{VLS}}| &= \left|\frac{g_k^T(g_k - t_k g_{k-1})}{-\lambda d_{k-1}^T g_{k-1} + (1-\lambda)\max\{0, g_k^T d_{k-1}\}}\right| \\
&\leq \frac{\|g_k\|(\|g_k - g_{k-1}\| + \|g_{k-1} - t_k g_{k-1}\|)}{\lambda|d_{k-1}^T g_{k-1}|} \\
&\leq \frac{\tilde{r}(L\tau + \|g_{k-1} - t_k g_{k-1}\|)}{\lambda|d_{k-1}^T g_{k-1}|} \leq \frac{\tilde{r}(L\tau + |\|g_k\| - \|g_{k-1}\||)}{(\lambda - 2\sigma)\|g_{k-1}\|^2} \\
&\leq \frac{\tilde{r}(L\tau + \|g_k - g_{k-1}\|)}{(\lambda - 2\sigma)\|g_{k-1}\|^2} \leq \frac{2L\tau\tilde{r}}{(\lambda - 2\sigma)r^2} = \frac{1}{2b}.
\end{aligned}$$

□

Lemma 3.4 *Suppose Assumption* (H) *holds. Consider any method of* (1.2)-(1.3), *where* $\beta_k \geq 0$, *and where* α_k *satisfies the strong Wolfe line search. If* β_k *has the property* $(*)$, *and* (2.5) *and* (3.4) *hold, then there exists a constant* $\tau > 0$, *for any* $\Delta \in Z^+$ *and* $k_0 \in Z^+$, *and for any* $k \geq k_0$ *such that*

$$\left|\Re^{\tau}_{k,\Delta}\right| > \frac{\Delta}{2},$$

where $\Re^{\tau}_{k,\Delta} \underline{\underline{\Delta}} \{i \in Z^+ : k \leq i \leq k + \Delta - 1, \|x_i - x_{i-1}\| \geq \tau\}$, $|\Re^{\tau}_{k,\Delta}|$ *denotes the number of* $\Re^{\tau}_{k,\Delta}$.

Proof This lemma plays an important role in proving the global convergences of PRP, HS, and LS conjugate gradient methods, and so on. It was originally proved in [17]. From Remark 2.1 and Lemma 3.3, it is easy to find that Algorithm 2.1 leads to the conclusion of Lemma 3.4. □

Theorem 3.1 *Suppose Assumption* (H) *holds. Let the sequences* $\{g_k\}$ *and* $\{d_k\}$ *be generated by Algorithm* 2.1. *If* β_k^{VLS} *has the property* $(*)$, *and* (2.5) *holds, then we obtain*

$$\liminf_{k \to +\infty} \|g_k\| = 0. \tag{3.5}$$

Proof Using mathematical induction. Suppose that (3.5) does not hold, which means that there exists $r > 0$ such that

$$\|g_k\| \geq r, \quad \text{for all } k \geq 1. \tag{3.6}$$

We also define $u_k = \frac{d_k}{\|d_k\|}$, then for all $l, k \in Z^+$ $(l \geq k)$, we have

$$\begin{aligned} x_l - x_{k-1} &= \sum_{i=k}^{l} \|x_i - x_{i-1}\| \cdot u_{i-1} \\ &= \sum_{i=k}^{l} \|s_{i-1}\| \cdot u_{k-1} + \sum_{i=k}^{l} \|s_{i-1}\| (u_{i-1} - u_{k-1}), \end{aligned} \tag{3.7}$$

where $s_{i-1} = x_i - x_{i-1}$.

From Assumption (H), we know that there exists a constant $\xi > 0$ such that

$$\|x\| \leq \xi, \quad \text{for } x \in V. \tag{3.8}$$

By (3.7), we have

$$\sum_{i=k}^{l} \|s_{i-1}\| \cdot u_{k-1} = (x_l - x_{k-1}) - \sum_{i=k}^{l} \|s_{i-1}\| (u_{i-1} - u_{k-1}). \tag{3.9}$$

Since (3.8) and (3.9) hold, we have

$$\sum_{i=k}^{l} \|s_{i-1}\| \leq 2\xi + \sum_{i=k}^{l} \|s_{i-1}\| \cdot \|u_{i-1} - u_{k-1}\|. \tag{3.10}$$

Let τ come from Lemma 3.4, and we define $\Delta = [8\xi/\tau]$, where $8\xi/\tau \leq \Delta < (8\xi/\tau) + 1$, and $\Delta \in Z^+$.

From Lemma 3.2, we know that there exists k_0 such that

$$\sum_{i\geq k_0} \|u_{i+1} - u_i\|^2 \leq \frac{1}{4\Delta}. \tag{3.11}$$

From the Cauchy-Schwarz inequality and (3.11), and letting $\forall i \in [k, k + \Delta - 1]$, we have

$$\begin{aligned} \|u_{i-1} - u_{k-1}\| &\leq \sum_{j=k}^{i-1} \|u_j - u_{j-1}\| \\ &\leq (i-k)^{\frac{1}{2}} \left(\sum_{j=k}^{i-1} \|u_j - u_{j-1}\|^2 \right)^{\frac{1}{2}} \\ &\leq \Delta^{\frac{1}{2}} \cdot \left(\frac{1}{4\Delta} \right)^{\frac{1}{2}} = \frac{1}{2}. \end{aligned} \tag{3.12}$$

From Lemma 3.4, we know that there exists $k \geq k_0$ such that

$$\left|\mathfrak{R}_{k,\Delta}^{\tau}\right| > \frac{\Delta}{2}. \tag{3.13}$$

By (3.10), (3.12), and (3.13), we have

$$2\xi \geq \frac{1}{2} \sum_{i=k}^{k+\Delta-1} \|s_{i-1}\| > \frac{\tau}{2} \left|\mathfrak{R}_{k,\Delta}^{\tau}\right| > \frac{\tau\Delta}{4}. \tag{3.14}$$

From (3.14), we have $\Delta < 8\xi/\tau$, which is a contradiction with the definition of Δ. Therefore,

$$\liminf_{k\to+\infty} \|g_k\| = 0.$$

Thus we complete the proof of Theorem 3.1. □

4 Numerical results

In this section, we compare the performance of Algorithm 2.1 with those of the PRP+ method [18] and the CG-DESCENT method [4] in the number of function evaluations and CPU time in seconds with the strong Wolfe line search. The test problems are some large-scaled unconstrained optimization problems in [19, 20]. The parameters in the line search are chosen as follows: $\delta = 0.01$, $\sigma = 0.1$. If $\|g_k\|_\infty \leq 10^{-6}$ is satisfied, we will terminate the program. All codes were written in Fortran 6.0 and run on a PC with 2.0 GHz CPU processor and 512 MB memory and Windows XP operation system.

The numerical results are reported in Table 1. The first column 'Problems' represents the problem's name in [19, 20]. 'Dim' denotes the dimension of the test problems. The detailed numerical results are listed in the form NF\CPU, where NF and CPU denote the number of function evaluations and CPU time in seconds, respectively.

We say that, in particular for the ith problem, the performance of the M1 method was better than the performance of M2 method, if the CPU time, or the number of function

Table 1 The numerical results of Algorithm 2.1, PRP+ method and CG-DESCENT method

Problems	Dim	Algorithm 2.1	PRP+ method	CG-DESCENT method
Extended Freudenstein & Roth	5,000	58\0.07	1,144\0.09	13,235\1.24
	10,000	12\0.01	426\0.07	23\0.01
Extended Trigonometric	5,000	30\0.07	61\0.08	79\0.10
	10,000	33\0.19	112\0.28	161\0.40
Extended Rosenbrock	5,000	41\0.02	67\0.02	60\0.02
	10,000	34\0.03	62\0.03	57\0.01
Extended White & Holst	5,000	40\0.02	53\0.02	45\0.02
	10,000	38\0.01	43\0.01	52\0.02
Extended Beale	5,000	15\0.01	26\0.02	24\0.00
	10,000	15\0.01	26\0.00	24\0.01
Extended Penalty	5,000	11\0.01	51\0.00	1,979\0.17
	10,000	18\0.02	36\0.02	50\0.02
Perturbed Quadratic	5,000	705\0.35	1,462\0.42	1,471\0.36
	10,000	1,353\0.88	2,059\1.19	2,014\0.95
Raydan 2	5,000	9\0.00	9\0.00	9\0.00
	10,000	9\0.02	9\0.02	9\0.02
Diagonal 2	5,000	432\0.44	987\0.67	699\0.45
	10,000	595\1.28	1,117\1.52	1,209\1.60
Generalized Tridiagonal 1	5,000	42\0.02	2,013\0.27	53\0.01
	10,000	71\0.14	578\0.15	707\0.19
Extended Tridiagonal 1	5,000	12\0.00	23\0.00	28\0.00
	10,000	12\0.01	28\0.02	29\0.01
Extended Three Expo Terms	5,000	8\0.01	21\0.01	15\0.02
	10,000	12\0.04	19\0.05	15\0.03
Generalized Tridiagonal 2	5,000	50\0.03	94\0.03	95\0.03
	10,000	62\0.05	97\0.06	77\0.03
Diagonal 4	5,000	8\0.00	8\0.00	8\0.00
	10,000	8\0.00	8\0.00	8\0.00
Diagonal 5	5,000	9\0.01	9\0.02	9\0.02
	10,000	9\0.03	9\0.03	9\0.03
Extended Himmelblau	5,000	18\0.00	35\0.00	16\0.00
	10,000	18\0.02	35\0.03	16\0.00
Generalized PSC1	5,000	1,886\3.20	633\0.60	17,679\14.59
	10,000	729\2.61	1,271\2.39	8,364\13.94
Extended PSC1	5,000	17\0.02	13\0.01	16\0.02
	10,000	17\0.03	15\0.02	16\0.03
Extended Powell	5,000	46\0.03	138\0.03	250\0.05
	10,000	74\0.06	88\0.04	311\0.09
Extended Block-Diagonal BD1	5,000	55\0.02	40\0.01	33\0.01
	10,000	53\0.06	47\0.05	46\0.05
Extended Maratos	5,000	71\0.03	132\0.03	103\0.02
	10,000	69\0.02	96\0.03	99\0.03
Quadratic Diagonal Perturbed	5,000	793\0.22	880\0.22	2,111\0.39
	10,000	1,549\0.62	1,303\0.66	2,966\1.16
Extended Wood	5,000	85\0.02	57\0.01	135\0.01
	10,000	84\0.04	65\0.03	116\0.05
Extended Hiebert	5,000	176\0.04	137\0.03	120\0.03
	10,000	173\0.05	137\0.05	114\0.03
QuadraticQF1	5,000	1,222\0.25	1,854\0.50	1,397\0.30
	10,000	1,396\0.82	1,864\0.97	2,545\1.06
Extended Quadratic Penalty QP2	5,000	45\0.05	80\0.06	76\0.06
	10,000	43\0.11	71\0.13	84\0.14
QuadraticQF2	5,000	1,167\0.40	1,620\0.44	1,613\0.36
	10,000	1,430\1.14	2,625\1.45	2,941\1.31
Extended EP1	5,000	6\0.00	6\0.00	6\0.00
	10,000	413\0.21	513\0.25	439\0.22
Extended Tridiagonal 2	5,000	875\0.23	2,436\0.27	857\0.11
	10,000	5,139\1.17	5,857\1.29	6,569\1.47
ARWHEAD	5,000	16\0.00	16\0.00	32\0.01
	10,000	11\0.00	14\0.00	11\0.00
NONDIA	5,000	15\0.00	15\0.00	17\0.00
	10,000	15\0.02	16\0.00	17\0.02

Table 1 (*Continued*)

Problems	Dim	Algorithm 2.1	PRP+ method	CG-DESCENT method
DQDRTIC	5,000	17\0.00	19\0.01	21\0.00
	10,000	26\0.01	25\0.00	23\0.01
DIXMAANA	5,000	11\0.02	12\0.02	16\0.00
	10,000	11\0.01	12\0.01	14\0.02
DIXMAANB	5,000	21\0.01	20\0.02	23\0.00
	10,000	21\0.02	20\0.02	23\0.03
DIXMAANC	5,000	22\0.01	24\0.01	28\0.02
	10,000	22\0.01	25\0.02	28\0.01
Broyden Tridiagonal	5,000	77\0.02	125\0.03	132\0.03
	10,000	76\0.04	121\0.08	110\0.05
Almost Perturbed Quadratic	5,000	1,156\0.30	1,479\0.39	1,448\0.31
	10,000	1,906\0.95	2,198\1.19	2,149\0.92
Tridiagonal Perturbed Quadratic	5,000	1,489\0.37	1,783\0.53	1,562\0.41
	10,000	1,140\1.15	1,879\1.11	2,477\1.25
EDENSCH	5,000	325\0.04	362\0.05	2,157\0.33
	10,000	1,106\0.26	1,492\0.45	1,594\0.47
VARDIM	5,000	34\0.00	46\0.02	46\0.00
	10,000	47\0.02	52\0.03	52\0.03
LIARWHD	5,000	24\0.01	29\0.02	34\0.00
	10,000	28\0.01	38\0.01	41\0.02
Diagonal 6	5,000	9\0.00	9\0.00	9\0.00
	10,000	9\0.01	9\0.02	9\0.01
DIXMAANG	5,000	1,186\0.42	715\0.39	660\0.33
	10,000	2,261\1.84	1,275\1.42	1,042\1.07
DIXMAANI	5,000	524\0.38	850\0.47	702\0.36
	10,000	658\1.15	1,266\1.37	985\1.00
DIXMAANJ	5,000	770\0.35	829\0.44	770\0.38
	10,000	1,823\1.19	1,108\1.14	1,108\1.09
DIXMAANK	5,000	1,510\0.52	715\0.41	812\0.41
	10,000	3,658\2.03	963\1.09	1,303\1.31
ENGVAL1	5,000	5,277\0.55	7,474\0.86	6,436\0.69
	10,000	7,380\1.72	7,196\1.62	21,402\4.56
COSINE	5,000	30\0.02	33\0.02	31\0.03
	10,000	30\0.03	34\0.04	31\0.03
DENSCHNB	5,000	10\0.01	14\0.02	13\0.02
	10,000	10\0.00	15\0.00	13\0.00
DENSCHNF	5,000	19\0.01	39\0.01	35\0.01
	10,000	19\0.02	29\0.02	31\0.02
SINQUAD	5,000	515\0.51	976\0.83	566\0.47
	10,000	2,011\2.61	2,116\3.59	5,989\10.08

evaluations, of the M1 method was smaller than the CPU time, or the number of iterations of the M2 method, respectively. In order to estimate the whole effect, we apply the performance profiles of Dolan and Moré [21] in CPU time. From Table 1, some CPU times are zero. In order to have a comprehensive evaluation of the M1 and M2 methods in CPU time, we take the average value of the CPU time for each method, and denote av(M1), av(M2). Then we take the CPU time of each problem plus the average value of av(M1) and av(M2). According to their description, the top curve is the method that solved the most problems in a time that was within a factor τ of the best time; see Figure 1 and Figure 2. Using the same method, we also test on the number of function evaluations; see Figure 3 and Figure 4.

Obviously, Algorithm 2.1 is competitive to the PRP+ method and the CG-DESCENT method in the number of function evaluations and CPU time. Thus, it is of great importance to study Algorithm 2.1.

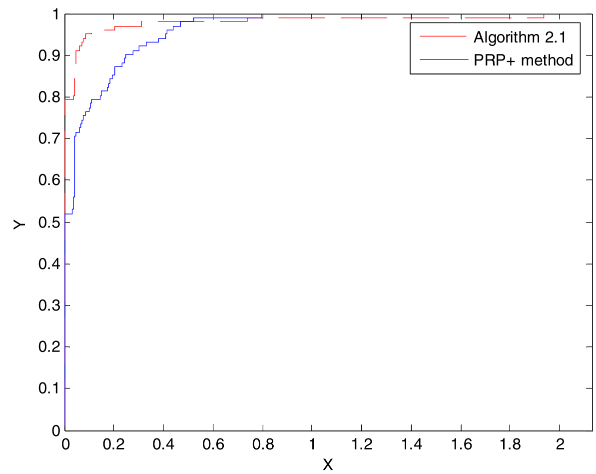

Figure 1 Performance profiles with respect to CPU time in seconds.

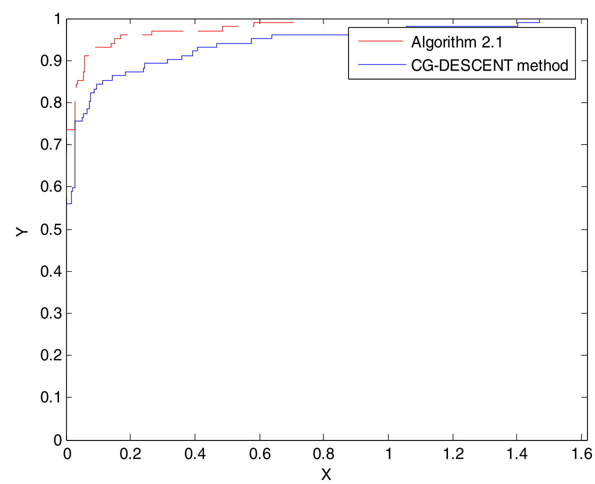

Figure 2 Performance profiles with respect to CPU time in seconds.

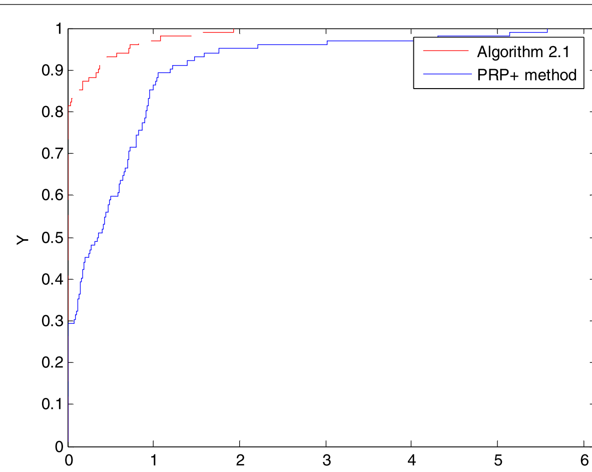

Figure 3 Performance profiles with respect to the numbers of iterations.

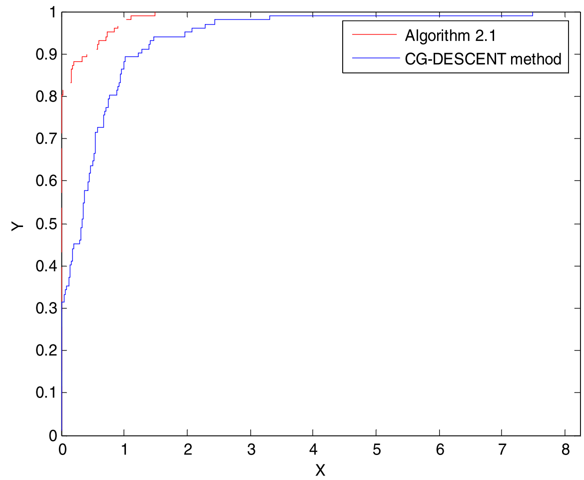

Figure 4 Performance profiles with respect to the numbers of iterations.

Competing interests
The author declares that she has no competing interests.

Acknowledgements
The author wishes to express their heartfelt thanks to the anonymous referees and the editor for their detailed and helpful suggestions for revising the manuscript.

Received: 19 March 2014 Accepted: 4 June 2014 Published: 18 Jul 2014

References

1. Polak, E, Ribière, G: Note sur la convergence de méthodes de directions conjuguées. Rev. Fr. Inform. Rech. Oper. **3**(16), 35-43 (1969)
2. Polak, BT: The conjugate gradient method in extreme problems. USSR Comput. Math. Math. Phys. **9**, 94-112 (1969)
3. Liu, Y, Storey, C: Efficient generalized conjugate gradient algorithms. Part 1: theory. J. Optim. Theory Appl. **69**, 129-137 (1992)
4. Hager, WW, Zhang, H: A new conjugate gradient method with guaranteed descent and an efficient line search. SIAM J. Optim. **16**, 170-192 (2005)
5. Powell, MJD: Nonconvex minimization calculations and the conjugate gradient method. In: Numerical Analysis. Lecture Notes in Mathematics, vol. 1066, pp. 122-141. Springer, Berlin (1984)
6. Hager, WW, Zhang, H: A survey of nonlinear conjugate gradient methods. Pac. J. Optim. **2**, 35-58 (2006)
7. Zhang, L, Zhou, W, Li, DH: A descent modified Polak-Ribière-Polyak conjugate gradient method and its global convergence. IMA J. Numer. Anal. **26**, 629-640 (2006)
8. Li, ZF, Chen, J, Deng, NY: A new conjugate gradient method and its global convergence properties. Math. Program. **78**, 375-391 (1997)
9. Liu, J: Convergence properties of a class of nonlinear conjugate gradient methods. Comput. Oper. Res. **40**, 2656-2661 (2013)
10. Liu, J, Du, X: Global convergence of a modified LS method. Math. Probl. Eng. **2012**, Article ID 910303 (2012)
11. Li, M, Chen, Y, Qu, A-P: Global convergence of a modified Liu-Storey conjugate gradient method. U.P.B. Sci. Bull., Ser. A **74**, 11-26 (2012)
12. Tang, C, Wei, Z, Li, G: A new version of the Liu-Storey conjugate gradient method. Appl. Math. Comput. **189**, 302-313 (2007)
13. Liu, J, Du, X, Wang, K: Convergence of descent methods with variable parameters. Acta Math. Appl. Sin. **33**, 222-230 (2010) (in Chinese)
14. Wei, Z, Yao, S, Liu, L: The convergence properties of some new conjugate gradient methods. Appl. Math. Comput. **183**, 1341-1350 (2006)
15. Zoutendijk, G: Nonlinear programming, computational methods. In: Abadie, J (ed.) Integer and Nonlinear Programming, pp. 37-86. North-Holland, Amsterdam (1970)
16. Li, ZF, Chen, J, Deng, NY: Convergence properties of conjugate gradient methods with Goldstein line searches. J. China Agric. Univ. **I**(4), 15-18 (1996)
17. Dai, YH, Yuan, Y: Nonlinear Conjugate Gradient Method. Shanghai Scientific & Technical Publishers, Shanghai (2000) (in Chinese)
18. Powell, MJD: Convergence properties of algorithms for nonlinear optimization. SIAM Rev. **28**, 487-500 (1986)
19. Bongartz, I, Conn, AR, Gould, NIM, Toint, PL: CUTE: constrained and unconstrained testing environments. ACM Trans. Math. Softw. **21**, 123-160 (1995)
20. Andrei, N: An unconstrained optimization test functions collection. Adv. Model. Optim. **10**, 147-161 (2008)
21. Dolan, ED, Moré, JJ: Benchmarking optimization software with performance profiles. Math. Program. **91**, 201-213 (2002)

10.1186/1029-242X-2014-248
Cite this article as: Wu: Global convergence of a modified conjugate gradient method. *Journal of Inequalities and Applications* 2014, 2014:248