Hindawi Publishing Corporation
Mathematical Problems in Engineering
Volume 2011, Article ID 463087, 22 pages
doi:10.1155/2011/463087

Research Article

Global Convergence of a Nonlinear Conjugate Gradient Method

Liu Jin-kui, Zou Li-min, and Song Xiao-qian

School of Mathematics and Statistics, Chongqing Three Gorges University, Chongqing 404000, China

Correspondence should be addressed to Liu Jin-kui, liujinkui2006@126.com

Received 5 March 2011; Revised 21 May 2011; Accepted 4 June 2011

Academic Editor: Piermarco Cannarsa

A modified PRP nonlinear conjugate gradient method to solve unconstrained optimization problems is proposed. The important property of the proposed method is that the sufficient descent property is guaranteed independent of any line search. By the use of the Wolfe line search, the global convergence of the proposed method is established for nonconvex minimization. Numerical results show that the proposed method is effective and promising by comparing with the VPRP, CG-DESCENT, and DL$^+$ methods.

1. Introduction

The nonlinear conjugate gradient method is one of the most efficient methods in solving unconstrained optimization problems. It comprises a class of unconstrained optimization algorithms which is characterized by low memory requirements and simplicity.

Consider the unconstrained optimization problem

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

where $f : R^n \to R$ is continuously differentiable, and its gradient g is available.

The iterates of the conjugate gradient method for solving (1.1) are given by

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

where stepsize α_k is positive and computed by certain 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 $g_k = \nabla f(x_k)$, and β_k is a scalar. Some well-known conjugate gradient methods include Polak-Ribière-Polyak (PRP) method [1, 2], Hestenes-Stiefel (HS) method [3], Hager-Zhang (HZ) method [4], and Dai-Liao (DL) method [5]. The parameters β_k of these methods are specified as follows:

$$\begin{gathered} \beta_k^{\mathrm{PRP}} = \frac{g_k^T (g_k - g_{k-1})}{\|g_{k-1}\|^2}, \\ \beta_k^{\mathrm{HS}} = \frac{g_k^T (g_k - g_{k-1})}{d_{k-1}^T (g_k - g_{k-1})}, \\ \beta_k^{\mathrm{HZ}} = \left(y_{k-1} - 2d_{k-1} \frac{\|y_{k-1}\|^2}{d_{k-1}^T y_{k-1}} \right)^T \frac{g_k}{d_{k-1}^T y_{k-1}}, \\ \beta_k^{\mathrm{DL}} = \frac{g_k^T y_{k-1}}{d_{k-1}^T y_{k-1}} - t \frac{g_k^T s_{k-1}}{d_{k-1}^T y_{k-1}}, \quad (t \geq 0), \end{gathered} \tag{1.4}$$

where $\|\cdot\|$ is the Euclidean norm and $y_{k-1} = g_k - g_{k-1}$. We know that if f is a strictly convex quadratic function, the above methods are equivalent in the case that an exact line search is used. If f is nonconvex, their behaviors may be further different.

In the past few years, the PRP method has been regarded as 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]). Powell [7] constructed an example which showed that the PRP method can cycle infinitely without approaching any stationary point even if an exact line search is used. This counterexample also indicates that the PRP method has a drawback that it may not globally be convergent when the objective function is nonconvex. Powell [8] suggested that the parameter β_k is negative in the PRP method and defined β_k as

$$\beta_k = \max\left\{0, \beta_k^{\mathrm{PRP}}\right\}. \tag{1.5}$$

Gilbert and Nocedal [9] considered Powell's suggestion and proved the global convergence of the modified PRP method for nonconvex functions under the appropriate line search. In addition, there are many researches on convergence properties of the PRP method (see [10–12]).

In recent years, much effort has been investigated to create new methods, which not only possess global convergence properties for general functions but also are superior to original methods from the computation point of view. For example, Yu et al. [13] proposed a new nonlinear conjugate gradient method in which the parameter β_k is defined on the basic of β_k^{PRP} such as

$$\beta_k^{\mathrm{VPRP}} = \begin{cases} \dfrac{\|g_k\|^2 - |g_k^T g_{k-1}|}{\nu|g_k^T d_{k-1}| + \|g_{k-1}\|^2} & \text{if } \|g_k\|^2 > |g_k^T g_{k-1}|, \\ 0, & \text{otherwise,} \end{cases} \tag{1.6}$$

where $\nu > 1$ (in this paper, we call this method as VPRP method). And they proved the global convergence of the VPRP method with the Wolfe line search. Hager and Zhang [4] discussed the global convergence of the HZ method for strong convex functions under the Wolfe line search and Goldstein line search. In order to prove the global convergence for general functions, Hager and Zhang modified the parameter β_k^{HZ} as

$$\beta_k^{\mathrm{MHZ}} = \max\left\{\beta_k^{\mathrm{HZ}}, \eta_k\right\}, \tag{1.7}$$

where

$$\eta_k = \frac{-1}{\|d_k\| \min\{\eta, \|g_k\|\}}, \quad \eta > 0. \tag{1.8}$$

The corresponding method of (1.7) is the famous CG-DESCENT method.

Dai and Liao [5] proposed a new conjugate condition, that is,

$$d_k^T y_{k-1} = -t g_k^T s_{k-1}, \quad (t \geq 0). \tag{1.9}$$

Under the new conjugate condition, they proved global convergence of the DL conjugate gradient method for uniformly convex functions. According to Powell's suggestion, Dai and Liao gave a modified parameter

$$\beta_k = \max\left\{\frac{g_k^T y_{k-1}}{d_{k-1}^T y_{k-1}}, 0\right\} - t\frac{g_k^T s_{k-1}}{d_{k-1}^T y_{k-1}}, \quad (t \geq 0). \tag{1.10}$$

The corresponding method of (1.10) is the famous DL^+ method. Under the strong Wolfe line search, they researched the global convergence of the DL^+ method for general functions. Zhang et al. [14] proposed a modified DL conjugate gradient method and proved its global convergence. Moreover, some researchers have been studying a new type of method called the spectral conjugate gradient method (see [15–17]).

This paper is organized as follows: in the next section, we propose a modified PRP method and prove its sufficient descent property. In Section 3, the global convergence of the method with the Wolfe line search is given. In Section 4, numerical results are reported. We have a conclusion in the last section.

2. Modified PRP Method

In this section, we propose a modified PRP conjugate gradient method in which the parameter β_k is defined on the basic of β_k^{PRP} as follows:

$$\beta_k^{\mathrm{MPRP}} = \begin{cases} \dfrac{\|g_k\|^2 - |g_k^T g_{k-1}|}{\max\{0, g_k^T d_{k-1}\} + \|g_{k-1}\|^2} & \text{if } \|g_k\|^2 \geq |g_k^T g_{k-1}| \geq m\|g_k\|^2, \\ 0, & \text{otherwise,} \end{cases} \tag{2.1}$$

in which $m \in (0,1)$. We introduce the modified PRP method as follows.

2.1. Modified PRP (MPRP) Method

Step 1. Set $x_1 \in R^n$, $\varepsilon \geq 0$, and $d_1 = -g_1$, if $\|g_1\| \leq \varepsilon$, then stop.

Step 2. Compute α_k by some inexact line search.

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 $k = k+1$, and go to Step 2.

In the convergence analyses and implementations of conjugate gradient methods, one often requires the inexact line search to satisfy the Wolfe line search or the strong Wolfe line search. The Wolfe line search is to find α_k such that

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

$$g(x_k + \alpha_k d_k)^T d_k \geq \sigma g_k^T d_k, \tag{2.3}$$

where $0 < \delta < \sigma < 1$. The strong Wolfe line search consists of (2.2) and the following strengthened version of (2.3):

$$\left| g(x_k + \alpha_k d_k)^T d_k \right| \leq -\sigma g_k^T d_k. \tag{2.4}$$

Moreover, in most references, we can see that the sufficient descent condition

$$g_k^T d_k \leq -c\|g_k\|^2, \quad c > 0 \tag{2.5}$$

is always given which plays a vital role in guaranteeing the global convergence properties of conjugate gradient methods. But, in this paper, d_k can satisfy (2.5) without any line search.

Theorem 2.1. *Consider any method* (1.2)-(1.3), *where* $\beta_k = \beta_k^{MPRP}$. *If* $g_k \neq 0$ *for all* $k \geq 1$, *then*

$$g_k^T d_k < -\|g_k\|^2, \quad \forall k \geq 1. \tag{2.6}$$

Proof. Multiplying (1.3) by g_k^T, we get

$$g_k^T d_k = -\|g_k\|^2 + \beta_k^{\mathrm{MPRP}} g_k^T d_{k-1}. \tag{2.7}$$

If $\beta_k^{\mathrm{MPRP}} = 0$, from (2.7), we know that the conclusion (2.6) holds. If $\beta_k^{\mathrm{MPRP}} \neq 0$, the proof is divided into two cases in the following.

Firstly, if $g_k^T d_{k-1} \leq 0$, then from (2.1) and (2.7), one has

$$\begin{aligned}
g_k^T d_k &= -\|g_k\|^2 + \frac{\|g_k\|^2 - |g_k^T g_{k-1}|}{\max\{0, g_k^T d_{k-1}\} + \|g_{k-1}\|^2} \cdot g_k^T d_{k-1} \\
&= -\|g_k\|^2 + \frac{\|g_k\|^2 - |g_k^T g_{k-1}|}{\|g_{k-1}\|^2} \cdot g_k^T d_{k-1} \\
&= \left(-\|g_k\|^2\right) \cdot \frac{\left(|g_k^T g_{k-1}|/\|g_k\|^2\right) \cdot g_k^T d_{k-1} - g_k^T d_{k-1} + \|g_{k-1}\|^2}{\|g_{k-1}\|^2} \\
&= \left(-\|g_k\|^2\right) \cdot \frac{\|g_{k-1}\|^2 - g_k^T d_{k-1}\left(1 - |g_k^T g_{k-1}|/\|g_k\|^2\right)}{\|g_{k-1}\|^2} \\
&\leq \left(-\|g_k\|^2\right) \cdot \frac{\|g_{k-1}\|^2}{\|g_{k-1}\|^2} = -\|g_k\|^2 < 0.
\end{aligned} \tag{2.8}$$

Secondly, if $g_k^T d_{k-1} > 0$, then from (2.7), we also have

$$g_k^T d_k < -\|g_k\|^2 + \frac{\|g_k\|^2 - |g_k^T g_{k-1}|}{g_k^T d_{k-1}} \cdot g_k^T d_{k-1} = -\left|g_k^T g_{k-1}\right| \leq -m\|g_k\|^2. \tag{2.9}$$

From the above, the conclusion (2.6) holds under any line search. □

3. Global Convergences of the Modified PRP Method

In order to prove the global convergence of the modified PRP method, we assume that the objective function $f(x)$ satisfies the following assumption.

Assumption H

(i) The level set $\Omega = \{x \in R^n \mid f(x) \leq f(x_1)\}$ is bounded, that is, there exists a positive constant $\xi > 0$ such that for all $x \in \Omega$, $\|x\| \leq \xi$.

(ii) In a neighborhood V of Ω, f is continuously differentiable and its gradient g is Lipchitz continuous, namely, there exists a constant $L > 0$ such that

$$\|g(x) - g(y)\| \leq L\|x - y\|, \quad \forall x, y \in V. \tag{3.1}$$

Under these assumptions on f, there exists a constant $\gamma > 0$ such that

$$\|g(x)\| \leq \gamma \quad \forall x \in \Omega. \tag{3.2}$$

The conclusion of the following lemma, often called the Zoutendijk condition, is used to prove the global convergence properties of nonlinear conjugate gradient methods. It was originally given by Zoutendijk [18].

Lemma 3.1. *Suppose that, Assumption H holds. Consider any iteration of* (1.2)-(1.3)*, where* d_k *satisfies* $g_k^T d_k < 0$ *for* $k \in N^+$ *and* α_k *satisfies the Wolfe line search, then*

$$\sum_{k\geq 1} \frac{\left(g_k^T d_k\right)^2}{\|d_k\|^2} < +\infty. \tag{3.3}$$

Lemma 3.2. *Suppose that Assumption H holds. Consider the method* (1.2)-(1.3)*, where* $\beta_k = \beta_k^{MPRP}$*, and* α_k *satisfies the Wolfe line search and* (2.6)*. If there exists a constant* $r > 0$*, such that*

$$\|g_k\| \geq r, \quad \forall k \geq 1, \tag{3.4}$$

then one has

$$\sum_{k\geq 2} \|u_k - u_{k-1}\|^2 < +\infty, \tag{3.5}$$

where $u_k = d_k/\|d_k\|$.

Proof. From (2.1) and (3.4), we get

$$g_k^T g_{k-1} \neq 0. \tag{3.6}$$

By (2.6) and (3.6), we know that $d_k \neq 0$ for each k.

Define the quantities

$$r_k = \frac{-g_k}{\|d_k\|}, \qquad \delta_k = \frac{\beta_k^{\mathrm{MPRP}} \|d_{k-1}\|}{\|d_k\|}. \tag{3.7}$$

By (1.3), one has

$$u_k = \frac{d_k}{\|d_k\|} = \frac{-g_k + \beta_k^{\mathrm{MPRP}} d_{k-1}}{\|d_k\|} = r_k + \delta_k u_{k-1}. \tag{3.8}$$

Since u_k is unit vector, we get

$$\|r_k\| = \|u_k - \delta_k u_{k-1}\| = \|\delta_k u_k - u_{k-1}\|. \tag{3.9}$$

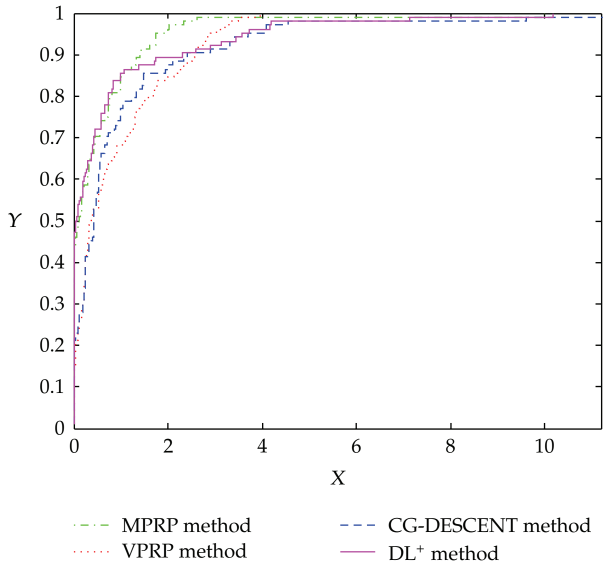

Figure 1: Performance profiles of the conjugate gradient methods with respect to CPU time.

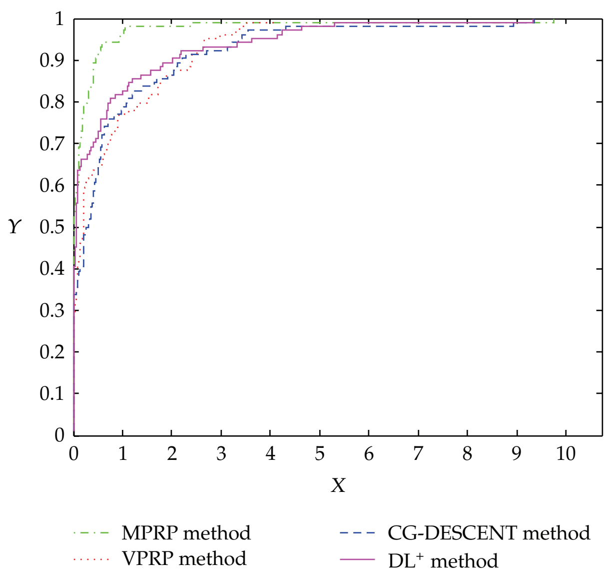

Figure 2: Performance profiles with respect to the number of iterations.

From $\delta_k \geq 0$ and the above equation, one has

$$\begin{aligned}\|u_k - u_{k-1}\| &\leq (1 + \delta_k)\|u_k - u_{k-1}\| \\ &= \|(1 + \delta_k)u_k - (1 + \delta_k)u_{k-1}\|\end{aligned}$$

$$\leq \|u_k - \delta_k u_{k-1}\| + \|\delta_k u_k - u_{k-1}\|$$
$$= 2\|r_k\|. \tag{3.10}$$

By (2.1), (3.4), and (3.6), one has

$$1 \geq \frac{\left|g_k^T g_{k-1}\right|^2}{\left\|g_k\right\|^2} > m. \tag{3.11}$$

From (3.3), (2.6), (3.4), and (3.11), one has

$$\begin{aligned} m^2 \sum_{k\geq 1, d_k \neq 0} \|r_k\|^2 &\leq \sum_{k\geq 1, d_k \neq 0} \left(\|r_k\|^2 \cdot \frac{\left|g_k^T g_{k-1}\right|^2}{\left\|g_k\right\|^2} \right) \\ &= \sum_{k\geq 1, d_k \neq 0} \frac{\left|g_k^T g_{k-1}\right|^2}{\|d_k\|^2} \\ &\leq \sum_{k\geq 1, d_k \neq 0} \frac{\left(g_k^T d_k\right)^2}{\|d_k\|^2} < +\infty, \end{aligned} \tag{3.12}$$

so

$$\sum_{k\geq 1, d_k \neq 0} \|r_k\|^2 < +\infty. \tag{3.13}$$

By (3.10) and the above inequality, one has

$$\sum_{k\geq 2} \|u_k - u_{k-1}\|^2 < +\infty. \tag{3.14}$$

□

Lemma 3.3. *Suppose that Assumption H holds. If* (3.4) *holds, then* β_k^{MPRP} *has property (*), that is,*

(1) *there exists a constant* $b > 1$*, such that* $|\beta_k^{MPRP}| \leq b$*,*

(2) *there exists a constant* $\lambda > 0$*, such that* $\|x_k - x_{k-1}\| \leq \lambda \Rightarrow |\beta_k^{MPRP}| \leq 1/2b$*.*

Proof. From Assumption (ii), we know that (3.2) holds. By (2.1), (3.2), and (3.4), one has

$$\left|\beta_k^{\mathrm{MPRP}}\right| \leq \frac{\left(\|g_k\| + \|g_{k-1}\|\right) \cdot \|g_k\|}{\left\|g_{k-1}\right\|^2} \leq \frac{2\gamma^2}{r^2} = b. \tag{3.15}$$

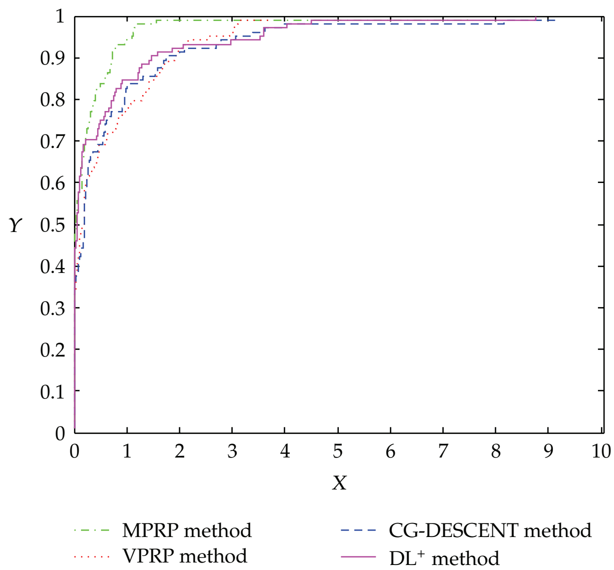

Figure 3: Performance profiles with respect to the number of function evaluations.

Define $\lambda = r^2/2L\gamma b$. If $||x_k - x_{k-1}|| \leq \lambda$, then from (2.1), (3.1), (3.2), and (3.4), one has

$$\begin{aligned} \left|\beta_k^{\text{MPRP}}\right| &\leq \frac{\|g_k\|^2 - g_k^T g_{k-1}}{\|g_{k-1}\|^2} = \frac{g_k^T(g_k - g_{k-1})}{\|g_{k-1}\|^2} \\ &\leq \frac{\|g_k\| \cdot \|g_k - g_{k-1}\|}{\|g_{k-1}\|^2} \leq \frac{\gamma L \lambda}{r^2} = \frac{1}{2b}. \end{aligned} \tag{3.16}$$

□

Lemma 3.4 (see [19]). *Suppose that Assumption H holds. Let* $\{x_k\}$ *and* $\{d_k\}$ *be generated by* (1.2)-(1.3)*, in which* α_k *satisfies the Wolfe line search and* (2.6)*. If* $\beta_k \geq 0$ *has the property (*) and* (3.4) *holds, then there exits* $\lambda > 0$*, for any* $\Delta \in Z^+$ *and* $k_0 \in Z^+$*, for all* $k \geq k_0$*, such that*

$$\left|\mathfrak{R}_{k,\Delta}^{\lambda}\right| > \frac{\Delta}{2}, \tag{3.17}$$

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

Theorem 3.5. *Suppose that Assumption H holds. Let* $\{x_k\}$ *and* $\{d_k\}$ *be generated by* (1.2)-(1.3)*, in which* α_k *satisfies the Wolfe line search and* (2.6)*,* $\beta_k = \beta_k^{MPRP}$*, then one has*

$$\liminf_{k \to +\infty} \|g_k\| = 0. \tag{3.18}$$

Table 1: The numerical results of the modified PRP method.

Problem	Dim	NI	NF	NG	CPU
ROSE	2	24	109	90	0.3651
FROTH	2	11	80	61	0.0594
BADSCP	2	26	227	210	0.2000
BADSCB	2	11	89	79	0.1085
BEALE	2	21	75	59	0.1449
HELIX	3	25	76	61	0.1754
BRAD	3	20	73	61	0.1380
GAUSS	3	3	8	6	0.0164
MEYER	3	1	1	1	0.0063
GULF	3	1	2	2	0.0173
BOX	3	1	1	1	0.0574
SING	4	67	263	228	0.5000
WOOD	4	33	150	117	0.2421
KOWOSB	4	57	222	195	0.4000
BD	4	26	127	96	0.1995
OSB1	5	1	1	1	0.0157
BIGGS	6	121	449	396	1.0000
OSB2	11	341	900	811	1.3000
JENSAM	6	12	49	32	0.0900
	7	13	56	35	0.1872
	8	11	53	30	0.1678
	9	12	65	38	0.1160
	10	26	133	94	0.2604
	11	NaN	NaN	NaN	NaN
VARDIM	3	4	40	26	0.0135
	5	6	57	38	0.0296
	6	5	65	43	0.0270
	8	7	72	47	0.0327
	9	7	78	50	0.0647
	10	7	81	52	0.0646
	12	7	90	58	0.0647
	15	8	92	60	0.0948
WATSON	5	59	200	167	0.2000
	6	387	1281	1134	1.4000
	7	1768	5834	5191	6.0000
	8	3934	13373	11920	14.0000
	10	4319	15102	13451	17.0000
	12	1892	6762	6007	9.0000
	15	1527	5552	4933	7.0000
	20	3001	11308	10107	19.0000
PEN2	5	111	439	393	0.4000
	10	185	845	752	1.5000
	15	154	774	679	0.5000
	20	178	989	889	0.6000
	30	123	610	534	0.4000
	40	147	700	617	0.5000
	50	152	744	651	1.2000
	60	163	813	720	0.7000

Table 1: Continued.

Problem	Dim	NI	NF	NG	CPU
PEN1	5	30	151	125	0.2742
	10	88	415	357	0.9000
	20	32	155	124	0.3349
	30	73	350	290	0.7000
	50	72	346	285	0.3000
	100	29	189	147	0.2458
	200	28	198	152	0.4759
	300	27	201	150	1.0464
TRIG	10	41	92	82	0.3817
	20	56	136	127	0.5634
	50	49	106	103	0.1949
	100	61	137	127	0.3857
	200	56	116	114	2.0205
	300	52	106	101	11.6394
	400	57	116	114	44.9734
	500	53	109	108	89.8125
ROSEX	100	26	123	103	0.2323
	200	26	123	103	0.2583
	300	26	123	103	0.3078
	400	26	123	103	0.4697
	500	26	123	103	0.6781
	1000	26	123	103	2.4474
	1500	26	123	103	5.3979
	2000	26	123	103	9.9364
SINGX	100	78	320	283	0.8000
	200	79	335	293	0.8000
	300	73	308	269	0.8000
	400	89	367	324	1.6000
	500	91	374	330	2.2000
	1000	93	385	342	8.0000
	1500	82	347	306	15.8000
	2000	80	341	299	28.4000
BV	200	1813	4326	4063	9.0000
	300	636	1501	1418	5.4000
	400	226	516	487	2.7000
	500	188	420	398	3.2000
	600	86	190	184	1.9000
	1000	21	40	37	0.9963
	1500	11	20	19	1.0900
	2000	2	6	5	0.5456
IE	200	6	13	7	0.3063
	300	6	13	7	0.6698
	400	6	13	7	1.1916
	500	6	13	7	1.8511
	600	6	13	7	2.6615
	1000	6	13	7	7.3635
	1500	6	13	7	16.6397
	2000	6	13	7	29.4927

Table 2: The numerical results of the VPRP method.

Problem	Dim	NI	NF	NG	CPU
TRID	200	35	81	74	0.3327
	300	36	83	75	0.3587
	400	37	83	75	0.3731
	500	35	78	73	0.4935
	600	36	80	76	0.6862
	1000	35	79	75	1.7180
	1500	36	84	79	4.0501
	2000	37	85	79	7.5866
ROSE	2	24	111	85	0.1585
FROTH	2	12	78	59	0.0698
BADSCP	2	99	394	336	0.4000
BADSCB	2	13	30	19	0.0448
BEALE	2	12	48	35	0.0402
HELIX	3	74	203	175	0.5000
BRAD	3	30	100	80	0.2027
GAUSS	3	3	7	4	0.0085
MEYER	3	1	1	1	0.0058
GULF	3	1	2	2	0.0052
BOX	3	1	1	1	0.0561
SING	4	101	341	289	0.7000
WOOD	4	174	482	417	0.6000
KOWOSB	4	71	234	203	0.3000
BD	4	42	161	125	0.2451
OSB1	5	1	1	1	0.0063
BIGGS	6	113	375	330	0.8000
OSB2	11	264	667	603	1.5000
JENSAM	6	9	33	17	0.0696
	7	11	39	17	0.1137
	8	10	42	19	0.0883
	9	17	90	57	0.1850
	10	17	124	84	0.1435
	11	6	76	46	0.1111
VARDIM	3	4	40	26	0.0290
	5	6	57	38	0.0203
	6	5	65	43	0.0273
	8	7	72	47	0.036
	9	7	78	50	0.0715
	10	7	81	52	0.0458
	12	7	90	58	0.0672
	15	8	92	60	0.0622
WATSON	5	193	535	473	1.4000
	6	342	1002	890	2.2000
	7	1451	4157	3678	5.0000
	8	6720	19530	17300	20.0000
	10	NaN	NaN	NaN	NaN
	12	3507	10432	9234	13.0000
	15	5271	15817	14006	24.0000
	20	NaN	NaN	NaN	NaN

Table 2: Continued.

Problem	Dim	NI	NF	NG	CPU
PEN2	5	129	499	434	1.0000
	10	90	379	328	0.3000
	15	434	1467	1298	2.2000
	20	941	2959	2612	3.0000
	30	771	2531	2193	3.0000
	40	248	978	860	1.6000
	50	952	2788	2511	3.0000
	60	927	2822	2435	4.0000
PEN1	5	32	151	124	0.3752
	10	90	383	324	0.8000
	20	28	160	121	0.1119
	30	76	334	279	0.3000
	50	64	344	280	0.5000
	100	23	160	122	0.2212
	200	21	174	128	0.4081
	300	28	192	143	1.0207
TRIG	10	36	82	71	0.4048
	20	56	125	114	0.6413
	50	45	93	85	0.3426
	100	58	120	113	0.4573
	200	64	135	128	2.0248
	300	52	102	99	12.4258
	400	60	132	125	52.5691
	500	59	127	116	101.6207
ROSEX	100	24	111	85	0.2798
	200	24	111	85	0.2808
	300	24	111	85	0.3136
	400	24	111	85	0.4271
	500	24	111	85	0.6181
	1000	24	111	85	2.1821
	1500	24	111	85	4.7644
	2000	24	111	85	8.8878
SINGX	100	169	562	483	0.6000
	200	199	676	576	1.4000
	300	365	1181	1031	3.7000
	400	627	2025	1756	9.0000
	500	129	431	367	2.5000
	1000	229	788	675	16.9000
	1500	100	329	280	16.0000
	2000	128	428	363	36.0000
BV	200	NaN	NaN	NaN	NaN
	300	7278	12990	12989	55.0000
	400	3837	6707	6706	42.0000
	500	1842	3236	3235	26.0000
	600	898	1562	1561	17.6000
	1000	133	232	231	6.2000
	1500	19	36	35	2.0484
	2000	2	6	5	0.5156

Table 2: Continued.

Problem	Dim	NI	NF	NG	CPU
IE	200	7	15	8	0.3596
	300	7	15	8	0.7986
	400	7	15	8	1.4202
	500	7	15	8	2.2147
	600	7	15	8	3.1811
	1000	7	15	8	8.8316
	1500	7	15	8	19.7838
	2000	7	15	8	34.7553
TRID	200	33	75	71	0.3758
	300	35	79	75	0.3654
	400	35	78	74	0.3912
	500	35	78	74	0.5188
	600	37	82	78	0.7388
	1000	34	76	72	1.6832
	1500	37	86	81	4.1950
	2000	37	86	80	7.5255

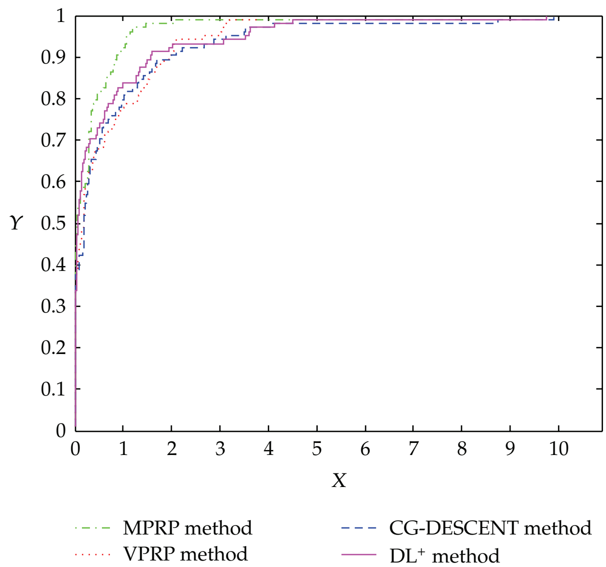

Figure 4: Performance profiles with respect to the number of gradient evaluations.

Proof. To obtain this result, we proceed by contradiction. Suppose that (3.18) does not hold, which means that there exists $r > 0$ such that

$$\|g_k\| \geq r, \quad \text{for } k \geq 1, \tag{3.19}$$

so, we know that Lemmas 3.2 and 3.4 hold.

Table 3: The numerical results of the CG-DESCENT method.

Problem	Dim	NI	NF	NG	CPU
ROSE	2	36	132	107	0.2371
FROTH	2	12	64	48	0.0462
BADSCP	2	40	213	189	0.2000
BADSCB	2	16	101	88	0.1212
BEALE	2	11	45	33	0.0405
HELIX	3	66	179	152	0.4397
BRAD	3	47	143	122	0.2292
GAUSS	3	3	10	8	0.0093
MEYER	3	1	1	1	0.0085
GULF	3	1	2	2	0.0068
BOX	3	1	1	1	0.0600
SING	4	62	198	164	0.3000
WOOD	4	103	298	247	0.5000
KOWOSB	4	77	222	192	0.4000
BD	4	53	204	162	0.3000
OSB1	5	1	1	1	0.0084
BIGGS	6	128	395	341	0.7000
OSB2	11	379	915	827	1.2000
JENSAM	6	NaN	NaN	NaN	NaN
	7	12	51	32	0.1114
	8	11	50	26	0.0844
	9	NaN	NaN	NaN	NaN
	10	5	59	34	0.0421
	11	21	148	105	0.1749
VARDIM	3	4	40	26	0.0246
	5	6	57	38	0.0174
	6	5	65	43	0.0266
	8	7	72	47	0.0226
	9	7	78	50	0.0323
	10	7	81	52	0.0495
	12	7	90	58	0.0610
	15	8	92	60	0.0583
WATSON	5	135	391	336	0.5000
	6	421	1186	1043	1.4000
	7	1822	5278	4655	6.0000
	8	2607	7589	6716	9.0000
	10	NaN	NaN	NaN	NaN
	12	3370	10111	8930	12.0000
	15	5749	17442	15368	27.0000
	20	5902	18524	16282	36.0000
PEN2	5	128	485	421	1.1000
	10	147	571	486	0.4000
	15	663	2262	1996	2.0000
	20	734	2452	2181	3.0000
	30	810	2438	2264	3.0000
	40	1488	4502	3960	5.0000
	50	744	2342	2056	2.0000
	60	755	2457	2188	3.0000

Table 3: Continued.

Problem	Dim	NI	NF	NG	CPU
PEN1	5	39	174	141	0.3298
	10	94	404	345	0.8000
	20	35	162	127	0.1220
	30	76	347	286	0.3000
	50	81	375	313	0.4000
	100	31	184	141	0.2434
	200	25	171	126	0.4106
	300	26	186	139	1.0160
TRIG	10	33	74	65	0.2781
	20	60	145	132	0.5098
	50	43	95	90	0.3620
	100	53	116	108	0.4474
	200	59	123	118	1.7555
	300	51	109	105	12.6411
	400	58	121	114	45.0506
	500	51	110	105	89.7073
ROSEX	100	36	124	99	0.1230
	200	34	125	102	0.1539
	300	35	133	107	0.3939
	400	31	121	100	0.5789
	500	31	129	106	0.8736
	1000	37	142	116	3.5602
	1500	34	132	106	7.4845
	2000	32	128	103	12.9238
SINGX	100	77	227	187	0.7000
	200	54	172	141	0.2376
	300	99	301	248	1.0000
	400	79	245	207	1.3000
	500	69	215	180	1.6000
	1000	101	322	271	8.6000
	1500	66	210	174	12.4000
	2000	69	210	173	23.1000
BV	200	NaN	NaN	NaN	NaN
	300	NaN	NaN	NaN	NaN
	400	4509	8044	8043	63.0000
	500	1635	2926	2925	34.0000
	600	925	1605	1604	24.5000
	1000	247	418	417	16.9000
	1500	18	38	37	3.0564
	2000	2	6	5	0.6883
IE	200	7	15	8	0.3546
	300	7	15	8	0.7927
	400	7	15	8	1.4039
	500	7	15	8	2.2022
	600	7	15	8	3.1297
	1000	7	15	8	8.6892
	1500	7	15	8	19.5607
	2000	7	15	8	34.7423

Table 3: Continued.

Problem	Dim	NI	NF	NG	CPU
TRID	200	31	70	58	0.2691
	300	33	73	65	0.2857
	400	32	72	61	0.4264
	500	34	76	71	0.6853
	600	35	78	74	0.9484
	1000	36	81	77	2.6056
	1500	36	84	79	5.8397
	2000	35	80	73	9.9491

We also define $u_k = d_k/\|d_k\|$, then for all $l, k \in Z^+$ $(l \geq k)$, one has

$$\begin{aligned} x_l - x_{k-1} &= \sum_{i=k}^{l} \|x_i - x_{i-1}\| \cdot u_{i-1} \\ &= \sum_{i=k}^{l} \|s_{i-1}\| \cdot u_{k-1} + \sum_{i=k}^{l} \|s_{i-1}\| (u_{i-1} - u_{k-1}), \end{aligned} \tag{3.20}$$

where $s_{i-1} = x_i - x_{i-1}$, that is,

$$\sum_{i=k}^{l} \|s_{i-1}\| \cdot u_{k-1} = (x_l - x_{k-1}) - \sum_{i=k}^{l} \|s_{i-1}\| (u_{i-1} - u_{k-1}). \tag{3.21}$$

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

$$\|x\| \leq \xi, \quad \text{for } x \in V. \tag{3.22}$$

From (3.21) and the above inequality, one has

$$\sum_{i=k}^{l} \|s_{i-1}\| \leq 2\xi + \sum_{i=k}^{l} \|s_{i-1}\| \cdot \|u_{i-1} - u_{k-1}\|. \tag{3.23}$$

Let Δ be a positive integer and $\Delta \in [8\xi/\lambda, 8\xi/\lambda + 1)$ where λ has been defined in Lemma 3.4. From Lemma 3.2, we know that there exists k_0 such that

$$\sum_{i \geq k_0} \|u_{i+1} - u_i\|^2 \leq \frac{1}{4\Delta}. \tag{3.24}$$

Table 4: The numerical results of the DL$^+$ method.

P	Dim	It	f_iter	Grade_iter	CPU
ROSE	2	25	110	87	0.1192
FROTH	2	9	65	50	0.0283
BADSCP	2	19	146	133	0.1003
BADSCB	2	11	56	47	0.0434
BEALE	2	11	50	39	0.0355
HELIX	3	50	144	122	0.2616
BARD	3	20	76	64	0.0824
GAUSS	3	2	5	3	0.0079
MEYER	3	1	1	1	0.0087
GULF	3	1	2	2	0.0062
BOX	3	1	1	1	0.0601
SING	4	91	295	250	0.5000
WOOD	4	148	402	349	0.8000
KOWOSB	4	76	236	204	0.4000
BD	4	67	208	177	0.3000
OSB1	5	1	1	1	0.0083
BIGGS	6	59	211	189	0.3000
OSB2	11	279	699	614	1.4000
JENSAM	6	9	35	19	0.0570
	7	9	35	17	0.1276
	8	NaN	NaN	NaN	NaN
	9	15	91	57	0.1148
	10	17	136	96	0.2084
	11	5	80	50	0.0882
VARDIM	3	4	40	26	0.0153
	5	6	57	38	0.0192
	6	5	65	43	0.0184
	8	7	72	47	0.0219
	9	7	78	50	0.0438
	10	7	81	52	0.0242
	12	7	90	58	0.0442
	15	8	92	60	0.0535
WASTON	5	62	208	173	0.2000
	6	346	1055	927	1.8000
	7	1247	3443	3085	4.0000
	8	2034	6104	5408	7.0000
	10	5973	17601	15699	21.0000
	12	3200	9291	8259	12.0000
	15	1865	5402	4790	8.0000
	20	6420	18320	16360	30.0000
PEN2	5	83	329	283	0.7000
	10	128	519	449	1.0000
	15	277	1040	922	0.9000
	20	290	1113	981	0.9000
	30	767	2231	2040	3.0000
	40	675	1929	1706	3.0000
	50	617	2223	1962	2.0000
	60	NaN	NaN	NaN	NaN

Table 4: Continued.

P	Dim	It	f_iter	Grade_iter	CPU
PEN1	5	33	165	139	0.2615
	10	78	353	298	0.7000
	20	24	119	94	0.0842
	30	73	367	305	0.3000
	50	66	356	293	0.6000
	100	22	154	117	0.1957
	200	27	178	136	0.4346
	300	29	195	147	1.0408
TRIG	10	34	82	71	0.2910
	20	59	140	129	0.5161
	50	46	94	88	0.3865
	100	55	120	111	0.4625
	200	56	116	111	1.6476
	300	52	110	105	12.3062
	400	56	115	113	44.8831
	500	54	121	116	97.7043
ROSEX	100	25	110	87	0.0859
	200	25	110	87	0.2019
	300	25	110	87	0.2697
	400	25	110	87	0.4169
	500	25	110	87	0.5990
	1000	25	110	87	2.2166
	1500	25	110	87	4.8426
	2000	25	110	87	9.1263
SINGX	100	113	384	329	1.0000
	200	117	397	341	0.5000
	300	215	735	634	2.1000
	400	113	384	329	1.6000
	500	110	367	313	2.2000
	1000	137	452	388	9.2000
	1500	152	494	420	22.1000
	2000	110	367	313	30.2000
BV	200	NaN	NaN	NaN	NaN
	300	NaN	NaN	NaN	NaN
	400	5624	8458	8457	49.0000
	500	3314	4854	4853	38.0000
	600	1618	2291	2290	25.0000
	1000	258	312	311	8.7000
	1500	18	30	29	1.7251
	2000	2	6	5	0.5129
IE	200	6	13	7	0.3075
	300	6	13	7	0.6731
	400	6	13	7	1.1939
	500	6	13	7	1.8608
	600	6	13	7	2.6785
	1000	6	13	7	7.3685
	1500	6	13	7	16.5794
	2000	6	13	7	29.5854

Table 4: Continued.

P	Dim	It	f_iter	Grade_iter	CPU
TRID	200	33	75	71	0.2606
	300	35	79	75	0.2866
	400	36	80	76	0.3653
	500	35	78	73	0.4857
	600	37	82	78	0.6950
	1000	34	76	72	1.6550
	1500	37	86	81	4.1380
	2000	37	86	80	7.4702

From the Cauchy-Schwartz inequality and (3.24), for all $i \in [k, k+\Delta-1]$, one has

$$\begin{aligned} \|u_{i-1} - u_{k-1}\| &\le \sum_{j=k}^{i-1} \|u_j - u_{j-1}\| \\ &\le (i-k)^{1/2} \left(\sum_{j=k}^{i-1} \|u_j - u_{j-1}\|^2 \right)^{1/2} \\ &\le \Delta^{1/2} \cdot \left(\frac{1}{4\Delta} \right)^{1/2} = \frac{1}{2}. \end{aligned} \tag{3.25}$$

By Lemma 3.4, we know that there exists $k \ge k_0$ such that

$$\left| \mathfrak{R}^{\lambda}_{k,\Delta} \right| > \frac{\Delta}{2}. \tag{3.26}$$

It follows from (3.23), (3.25), and (3.26) that

$$\frac{\lambda\Delta}{4} < \frac{\lambda}{2} \left| \mathfrak{R}^{\lambda}_{k,\Delta} \right| < \frac{1}{2} \sum_{i=k}^{k+\Delta-1} \|s_{i-1}\| \le 2\xi. \tag{3.27}$$

From (3.27), one has $\Delta < 8\xi/\lambda$, which is a contradiction with the definition of Δ. Hence,

$$\liminf_{k \to +\infty} \|g_k\| = 0, \tag{3.28}$$

which completes the proof. □

4. Numerical Results

In this section, we compare the modified PRP conjugate gradient method, denoted the MPRP method, to VPRP method, CG-DESCENT method, and DL^+ method under the strong Wolfe line search about problems [20] with the given initial points and dimensions. The parameters are chosen as follows: $\delta = 0.01$, $\sigma = 0.1$, $v = 1.25$, $\eta = 0.01$, and $t = 0.1$. If $\|g_k\| \le 10^{-6}$ is satisfied, we will stop the program. The program will be also stopped if the number of iteration is more than ten thousands. All codes were written in Matlab 7.0 and run on a PC with 2.0 GHz CPU processor and 512 MB memory and Windows XP operation system.

The numerical results of our tests with respect to the MPRP method, VPRP method, CG-DESCENT method, and DL^+ method are reported in Tables 1, 2, 3, 4, respectively. In the tables, the column "Problem" represents the problem's name in [20], and "CPU," "NI," "NF," and "NG" denote the CPU time in seconds, the number of iterations, function evaluations, gradient evaluations, respectively. "Dim" denotes the dimension of the tested problem. If the limit of iteration was exceeded, the run was stopped, and this is indicated by NaN.

In this paper, we will adopt the performance profiles by Dolan and Moré [21] to compare the MPRP method to the VPRP method, CG-DESCENT method, and DL^+ method in the CPU time, the number of iterations, function evaluations, and gradient evaluations performance, respectively (see Figures 1, 2, 3, 4). In figures,

$$X = \tau \longmapsto \frac{1}{n_p}\text{size}\left\{p \in P : \log_2\left(r_{p,s}\right) \le \tau\right\}, \qquad Y = P\left\{r_{p,s} \le \tau : 1 \le s \le n_s\right\}. \tag{4.1}$$

Figures 1–4 show the performance of the four methods relative to CPU time, the number of iterations, the number of function evaluations, and the number of gradient evaluations, respectively. For example, the performance profiles with respect to CPU time means that for each method, we plot the fraction P of problems for which the method is within a factor τ of the best time. The left side of the figure gives the percentage of the test problems for which a method is the fastest; the right side gives the percentage of the test problems that are successfully solved by each of the methods. The top curve is the method that solved of the most problems in a time that was within a factor τ of the best time.

Obviously, Figure 1 shows that MPRP method outperforms VPRP method, CG-DESCENT method, and DL^+ method for the given test problems in the CPU time. Figures 2–4 show that the MPRP method also has the best performance with respect to the number of iterations and function and gradient evaluations since it corresponds to the top curve. So, the MPRP method is computationally efficient.

5. Conclusions

We have proposed a modified PRP method on the basic of the PRP method, which can generate sufficient descent directions with inexact line search. Moreover, we proved that the proposed modified method converge globally for general nonconvex functions. The performance profiles showed that the proposed method is also very efficient.

Acknowledgments

The authors wish to express their heartfelt thanks to the referees and Professor Piermarco Cannarsa for their detailed and helpful suggestions for revising the paper. This work was supported by The Nature Science Foundation of Chongqing Education Committee (KJ091104) and Chongqing Three Gorge University (09ZZ-060).

References

[1] E. Polak and G. Ribire, "Note sur la xonvergence de directions conjugees," *Rev Francaise Informat Recherche Operatinelle 3e Annee*, vol. 16, pp. 35–43, 1969.

[2] B. T. Polak, "The conjugate gradient method in extreme problems," *USSR Computational Mathematics and Mathematical Physics*, vol. 9, pp. 94–112, 1969.

[3] M. Al-Baali, "Descent property and global convergence of the Fletcher-Reeves method with inexact line search," *IMA Journal of Numerical Analysis*, vol. 5, no. 1, pp. 121–124, 1985.

[4] W. W. Hager and H. Zhang, "A new conjugate gradient method with guaranteed descent and an efficient line search," *SIAM Journal on Optimization*, vol. 16, no. 1, pp. 170–192, 2005.

[5] Y. H. Dai and L. Z. Liao, "New conjugacy conditions and related nonlinear conjugate gradient methods," *Applied Mathematics and Optimization*, vol. 43, no. 1, pp. 87–101, 2001.

[6] W. W. Hager and H. Zhang, "A survey of nonlinear conjugate gradient methods," *Pacific Journal of Optimization*, vol. 2, no. 1, pp. 35–58, 2006.

[7] M. J. D. Powell, "Nonconvex minimization calculations and the conjugate gradient method," in *Numerical Analysis (Dundee, 1983)*, vol. 1066 of *Lecture Notes in Mathematics*, pp. 122–141, Springer, Berlin, Germany, 1984.

[8] M. J. D. Powell, "Convergence properties of algorithms for nonlinear optimization," *SIAM Review*, vol. 28, no. 4, pp. 487–500, 1986.

[9] J. C. Gilbert and J. Nocedal, "Global convergence properties of conjugate gradient methods for optimization," *SIAM Journal on Optimization*, vol. 2, no. 1, pp. 21–42, 1992.

[10] L. Zhang, W. Zhou, and D. Li, "A descent modified Polak-Ribière-Polyak conjugate gradient method and its global convergence," *IMA Journal of Numerical Analysis*, vol. 26, no. 4, pp. 629–640, 2006.

[11] Z. Wei, G. Y. Li, and L. Qi, "Global convergence of the Polak-Ribière-Polyak conjugate gradient method with an Armijo-type inexact line search for nonconvex unconstrained optimization problems," *Mathematics of Computation*, vol. 77, no. 264, pp. 2173–2193, 2008.

[12] L. Grippo and S. Lucidi, "A globally convergent version of the Polak-Ribière conjugate gradient method," *Mathematical Programming*, vol. 78, no. 3, pp. 375–391, 1997.

[13] G. Yu, Y. Zhao, and Z. Wei, "A descent nonlinear conjugate gradient method for large-scale unconstrained optimization," *Applied Mathematics and Computation*, vol. 187, no. 2, pp. 636–643, 2007.

[14] J. Zhang, Y. Xiao, and Z. Wei, "Nonlinear conjugate gradient methods with sufficient descent condition for large-scale unconstrained optimization," *Mathematical Problems in Engineering*, vol. 2009, Article ID 243290, 16 pages, 2009.

[15] Y. Xiao, Q. Wang, and D. Wang, "Notes on the Dai-Yuan-Yuan modified spectral gradient method," *Journal of Computational and Applied Mathematics*, vol. 234, no. 10, pp. 2986–2992, 2010.

[16] Z. Wan, Z. Yang, and Y. Wang, "New spectral PRP conjugate gradient method for unconstrained optimization," *Applied Mathematics Letters*, vol. 24, no. 1, pp. 16–22, 2011.

[17] L. Zhang, W. Zhou, and D. Li, "Global convergence of a modified Fletcher-Reeves conjugate gradient method with Armijo-type line search," *Numerische Mathematik*, vol. 104, no. 4, pp. 561–572, 2006.

[18] G. Zoutendijk, "Nonlinear programming, computational methods," in *Integer and Nonlinear Programming*, J. A. badie, Ed., pp. 37–86, North-Holland, Amsterdam, Netherlands, 1970.

[19] Y. H. Dai and Y. Yuan, *Nonlinear Conjugate Gradient Method*, vol. 10, Shanghai Scientific & Technical, Shanghai, China, 2000.

[20] J. J. Moré, B. S. Garbow, and K. E. Hillstrom, "Testing unconstrained optimization software," *Association for Computing Machinery Transactions on Mathematical Software*, vol. 7, no. 1, pp. 17–41, 1981.

[21] E. D. Dolan and J. J. Moré, "Benchmarking optimization software with performance profiles," *Mathematical Programming*, vol. 91, no. 2, pp. 201–213, 2002.