A Modified Hestenes-Stiefel Conjugate Gradient Algorithm for Large-Scale Optimization

Article *in* Numerical Functional Analysis and Optimization · August 2013

DOI: 10.1080/01630563.2013.777350

2 authors:

Gonglin Yuan
University of Technology Sydney

Maojun Zhang
guil university

A Modified Hestenes-Stiefel Conjugate Gradient Algorithm For Large-Scale Optimization *

Gonglin Yuan † Maojun Zhang‡§

Abstract. Mathematical programming is a rich and well-developed area in operations research. Nevertheless, there remain many challenging problems in this area, one of which is the large-scale optimization problem. In this paper, a modified Hestenes and Stiefel (HS) conjugate gradient (CG) algorithm with a nonmonotone line search technique is presented. This algorithm possesses information about not only the gradient value but also the function value. Moreover, the sufficient descent condition holds without any line search. The global convergence is established for nonconvex functions under suitable conditions. Numerical results show that the proposed algorithm is advantageous to existing CG methods for large-scale optimization problems.

Key Words. conjugate gradient; sufficient descent; global convergence.
AMS 2000 subject classifications. 90C26.

1. Introduction

Consider the problem

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

where $f:\Re^n\to\Re$ is continuously differentiable. The nonlinear CG method is one of the most effective line search methods for (1.1) due to its simplicity and its very low memory requirement. The iterative formula of the CG methods is defined by

$$x_{k+1}=x_k+\alpha_k d_k,\ k=1,2,\cdots \tag{1.2}$$

where x_k is the current iteration point, $\alpha_k>0$ is a steplength, and d_k is the search direction determined by

$$d_{k+1}=\begin{cases}-g_{k+1}+\beta_k d_k, & \text{if}\quad k\ge 1\\ -g_{k+1}, & \text{if}\quad k=0,\end{cases} \tag{1.3}$$

where g_{k+1} is the gradient of $f(x)$ at the point x_{k+1} and $\beta_k\in\Re$ is a scalar which determines different CG methods (see [7, 8, 11, 12, 18, 23, 26, 32]). From the literature, one hopes to find the steplength α_k using the following weak Wolfe-Powell (WWP) line search

$$f(x_k+\alpha_k d_k)\le f_k+\delta\alpha_k g_k^T d_k \tag{1.4}$$

and

$$g(x_k+\alpha_k d_k)^T d_k\ge\sigma g_k^T d_k, \tag{1.5}$$

where $\delta\in(0,1/2)$, and $\sigma\in(\delta,1)$. Under the WWP conditions, some formulae have the global convergence property but do not outperform than the well-known Polak-Ribière-Polyak (PRP) method in numerical computation. In the past few years, many efforts have been put to develop new CG formulae that

*This work is supported by Guangxi NSF (Grant No. 2012GXNSFAA053002 and 2012GXNSFAA053013), China NSF (Grant No. 11261006, 11161003, 71101033 and 71001015), and the Scientific Research Foundation of GuangXi University (Grant No. XJZ110632).

†College of Mathematics and Information Science, Guangxi University, Nanning, Guangxi, 530004, P.R. China. E-mail address: glyuan@gxu.edu.cn.
‡Shanghai Advanced Institute of Finance, Shanghai Jiao Tong University, Shanghai, 200030, P.R. China.
§Guilin University of Eelectronic Techonology, School of Mathematic and Computing Science, Guilin, Guangxi, 541004, P.R. China. E-mail: zhang1977108@sina.com.

possesses both global convergence property for general functions and good numerical performance (see [8, 13] in detail). The following sufficient descent condition

$$g_k^T d_k \leq -c\|g_k\|^2,\ \forall\ k \geq 1 \textit{ and some constant } c > 0 \tag{1.6}$$

is often used to analyze the global convergence of the CG method with the inexact line search techniques. Wei et al. pointed out that any new CG method should at least satisfy one of the following conditions [25]:

(i) The method with the WWP line search rule (or other line search rules) has some strongly convergent properties. The method with the WWP line search rule (or other line search rules) may at least generate a descent direction at each iteration, and converge globally.

(ii) The average performance on the numerical computation of the method with WWP line search rule (or others) should not be more inferior to that of the PRP method.

In recent years, many nCG formulae which possess the sufficient descent property (1.6) without any line search have been proposed (see [14, 16, 17, 21, 28, 29, 30, 31, 35] etc.). For instance, Yuan [29] proposed another modified HS formula defined by

$$\beta_k^{MHS} = \beta_k^{HS} - \min\{\beta_k^{HS}, \frac{\mu\|y_k\|^2}{(d_k^T y_k)^2} g_{k+1}^T d_k\}, \tag{1.7}$$

where $\mu > \frac{1}{4}$ is a constant, $y_k = g_{k+1} - g_k$, and $\beta_k^{HS} = \frac{g_{k+1}^T y_k}{d_k^T y_k}$. This method possesses the sufficient descent property and the global convergence with WWP line search. Based on Dai and Liao [6], Hager and Zhang ($HZ+$) proposed a new CG method (see [16, 17])

$$\beta_k^{HZ^+} = \max\{\beta_k^{HZ}, \zeta_k\}, \tag{1.8}$$

where $\beta_k^{HZ} = \frac{g_{k+1}^T (y_k - 2\frac{\|y_k\|^2}{s_k^T y_k} s_k)}{d_k^T y_k}$ with $s_k = x_{k+1} - x_k$ and $y_k = g_{k+1} - g_k$, and $\zeta_k = \frac{-1}{\|d_k\|, \min\{\zeta, \|g_k\|\}}$ with constant $\zeta > 0$. This method can be regarded as a modified HS method which also guarantees that d_k provides a descent direction and possesses global convergence with WWP line search. Numerical results show that this method is better than existing CG methods (such as the PRP, the PRP+, the HS, and the DY, etc.) and the limited memory BFGS method (see [16, 17] in detail). Today, this method (1.8) is considered to be one of the most effective algorithms. In this paper, we design a new CG method that has the following attributes.

• The given method possesses the sufficient descent property without any line search technique.

• Numerical results show that the given method is competitive over the HZ^+ method and other CG methods.

• The global convergence of the new method is established for nonconvex functions.

In the next section, the motivation and the algorithm are presented. The sufficient descent property and the global convergence of the new method are proven in Section 3. In Section 4, numerical results are reported. The last section contains concluding remarks.

2. Motivation and Algorithm

In this section, we present motivations based on the BFGS formulas and the line search technique, respectively.

2.1. Motivations based on BFGS formula

It is well known that the BFGS method is one of the most effective methods for unconstrained optimization problems. This method has also been shown to perform well(see [2, 3, 4, 5, 9, 19, 20, 22] etc.). Wei, Yu, Yuan, and Lian [27] presented a new BFGS update method generated by Taylor's formula as follows:

$$B_{k+1} = B_k - \frac{B_k s_k s_k^T B_k}{s_k^T B_k s_k} + \frac{y_k^* {y_k^*}^T}{s_k^T y_k^*}, \tag{2.1}$$

where $y_k^* = y_k + \frac{\rho_k}{\|s_k\|^2} s_k$ and $\rho_k = 2[f(x_k) - f(x_k + \alpha_k d_k)] + (g(x_k + \alpha_k d_k) + g(x_k))^T s_k$. Under the assumption that the objective functions are uniformly convex, the superlinear convergence of the new BFGS algorithm is given with the WWP line search. From the quasi-Newton equation

$$B_{k+1} s_k = y_k^* \tag{2.2}$$

which contains not only gradient value information but also function value information at the present and the previous steps, one may argue that the resulting methods would outperform the original BFGS method. Supporting this argument, numerical computations show that this method is better than the normal BFGS method (see [24, 27] for detail). Furthermore, some theoretical advantages of the new quasi-Newton equation (2.2) have been discussed (see [24] in detail). It can be seen that if the objective function f is uniformly convex, then

$$s_k^T y_k^* = s_k^T y_k + 2[f_k - f_{k+1}] + [g_{k+1} + g_k]^T s_k = 2 s_k^T g_{k+1} + 2(f_k - f_{k+1}) > 0$$

holds, where the last inequality is due to the uniform convexity of f. Hence, the updating formula (2.1) can ensure the positive definiteness of the matrix B_k for uniformly convex functions, and the superlinear convergence of this method has been established. However, if f is a generally convex function, then $s_k^T y_k^*$ may equal to zero. In this case, the updating is not positive definite anymore. Moreover, the global convergence and the superlinear convergence are still open for the generally convex function. Considering this, Yuan and Wei [33] define the quasi-Newton equation as:

$$B_{k+1} s_k = y_k^m, \tag{2.3}$$

where $y_k^m = y_k + \frac{\max\{\rho_k, 0\}}{\|s_k\|^2} s_k$, and

$$B_{k+1} = B_k - \frac{B_k s_k s_k^T B_k}{s_k^T B_k s_k} + \frac{y_k^m (y_k^m)^T}{s_k^T y_k^m}, \tag{2.4}$$

which ensures that B_{k+1} inherits the positive definiteness of B_k for the generally convex function. The global convergence and the superlinear convergence have been established for generally convex functions. Numerical results confirm the usefulness of this method.

Motivated by the above discussions and the CG formula (1.7), the modified HS formula replaces y_k by y_k^m. Accordingly, the new CG formula is defined as

$$d_{k+1} = \begin{cases} -g_{k+1} + \beta_k^m d_k, & \text{if } k \geq 1 \\ -g_{k+1}, & \text{if } k = 0, \end{cases} \tag{2.5}$$

where $\beta_k^m = \frac{g_{k+1}^T y_k^m}{d_k^T y_k^m} - \min\{\frac{g_{k+1}^T y_k^m}{d_k^T y_k^m}, \frac{\mu \|y_k^m\|^2}{(d_k^T y_k^m)^2} g_{k+1}^T d_k\}$ and $\mu > \frac{1}{4}$. Based on the new method (2.5), we state our algorithm in the following section.

2.2. Algorithm

Zhang and Hager [34] presented a new nonmonotone line search technique defined by:

$$f(x_k + \alpha_k d_k) \leq C_k + \delta \alpha_k g(x_k)^T d_k, \tag{2.6}$$

$$g(x_k + \alpha_k d_k)^T d_k \geq \sigma g(x_k)^T d_k, \tag{2.7}$$

where $0 < \delta < \sigma < 1$, $C_{k+1} = \frac{\eta_k Q_k C_k + f(x_k+\alpha_k d_k)}{Q_{k+1}}$, $Q_{k+1} = \eta_k Q_k + 1$, $\eta_k \in [\eta_{min}, \eta_{max}]$, $0 \le \eta_{min} \le \eta_{max} \le 1$, $C_1 = f(x_1)$, and $Q_1 = 1$. It is not difficult to see that C_{k+1} is a convex combination of C_k and $f(x_{k+1})$. Since $C_1 = f(x_1)$, it follows that C_k is a convex combination of the function values $f(x_1), f(x_2), \cdots, f(x_k)$. The choice of η_k controls the degree of nonmonotonicity. If $\eta_k = 0$ for each k, then the line search is the usual monotone Wolfe or Armijo line search. If $\eta_k = 1$ for each k, then $C_k = A_k$, where

$$A_k = \frac{1}{k}\sum_{i=1}^{k} f(x_i)$$

is the average function value. Numerical results show that this technique is better than standard nonmonotone techniques. Considering the efficiency of this technique, we will use this technique to find steplength α_k in our algorithm.

Algorithm 1(Nonmonotone HS conjugate gradient method)

Step 0: Choose an initial point $x_1 \in \Re^n$, $\varepsilon \in (0,1)$, $0 < \delta < \sigma < 1$, $0 \le \eta_{min} \le \eta_{max} < 1$. Set $d_1 = -g_1 = -\nabla f(x_1)$, $Q_1 = 1$, $C_1 = f(x_1)$, $k := 1$.

Step 1: If $\|g_k\| \le \varepsilon$, then stop; Otherwise, proceed to the next step.

Step 2: Compute step size α_k by line search rules (2.6) and (2.7).

Step 3: Let $x_{k+1} = x_k + \alpha_k d_k$. If $\|g_{k+1}\| \le \varepsilon$, then stop.

Step 4: Calculate the search direction by (2.5).

Step 5: Set $k := k+1$, and proceed to Step 2.

In the following section, we show that the given algorithm possesses the sufficient descent property without any line search technique and the global convergence for the general functions.

3. The sufficient descent property and the global convergence

Lemma 3.1 *Given (2.5), for $k \ge 1$, there exists a constant $c > 0$ such that*

$$d_{k+1}^T g_{k+1} \le -c\|g_{k+1}\|^2 \tag{3.1}$$

and

$$d_k^T y_k^m \ge c(1-\sigma)\|g_k\|^2. \tag{3.2}$$

Proof. If $k = 1$, then $g_1^T d_1 = -\|g_1\|^2$, (3.1) holds. For all $k \ge 1$, we assume that (3.1) holds, then For $k+1$, by (2.5), we have

$$\begin{aligned} g_{k+1}^T d_{k+1} &= -\|g_{k+1}\|^2 + \beta_k^m d_k^T g_{k+1} \\ &= -\|g_{k+1}\|^2 + (\frac{g_{k+1}^T y_k^m}{d_k^T y_k^m} - \min\{\frac{g_{k+1}^T y_k^m}{d_k^T y_k^m}, \frac{\mu\|y_k^m\|^2}{(d_k^T y_k^m)^2} g_{k+1}^T d_k\}) d_k^T g_{k+1}. \end{aligned} \tag{3.3}$$

Using the definition of y_k^m, (3.1), and the relation (2.7), we obtain

$$d_k^T y_k^m = d_k^T (y_k + \frac{\max\{\rho_k,\ 0\}}{\|s_k\|^2} s_k) \ge d_k^T y_k = d_k^T (g_{k+1} - g_k) \ge -(1-\sigma) g_k^T d_k > 0. \tag{3.4}$$

Denote $u = \frac{\sqrt{d_k^T y_k^m}}{\sqrt{2\mu}} g_{k+1}$, $v = \frac{\sqrt{2\mu} g_{k+1}^T d_k}{\sqrt{d_k^T y_k^m}} y_k^m$. We discuss (3.3) by the following two cases.

Case i. $\frac{g_{k+1}^T y_k^m}{d_k^T y_k^m} < \frac{\mu\|y_k^m\|^2}{(d_k^T y_k^m)^2} g_{k+1}^T d_k$. By (3.3), we get $g_{k+1}^T d_{k+1} = -\|g_{k+1}\|^2$.

Case ii. $\frac{g_{k+1}^T y_k^m}{d_k^T y_k^m} \ge \frac{\mu\|y_k^m\|^2}{(d_k^T y_k^m)^2} g_{k+1}^T d_k$. The equation (3.3) can be rewritten as

$$g_{k+1}^T d_{k+1} = -\|g_{k+1}\|^2 + (\frac{g_{k+1}^T y_k^m}{d_k^T y_k^m} - \frac{\mu\|y_k^m\|^2}{(d_k^T y_k^m)^2} g_{k+1}^T d_k\}) d_k^T g_{k+1}$$

$$
\begin{aligned}
&= \frac{d_k^T g_{k+1} g_{k+1}^T y_k^m - \|g_{k+1}\|^2 d_k^T y_k^m - \frac{\mu \|y_k^m\|^2}{d_k^T y_k^m}(g_{k+1}^T d_k)^2}{d_k^T y_k^m} \\
&= \frac{u^T v - \frac{1}{2}(\|u\|^2 + \|v\|^2)}{d_k^T y_k^m} + \frac{-(1-\frac{1}{4\mu})\|g_{k+1}\|^2 d_k^T y_k^m}{d_k^T y_k^m} \\
&\leq -(1-\frac{1}{4\mu})\|g_{k+1}\|^2,
\end{aligned}
$$

where the last inequality is due to the inequality $u^T v \leq \frac{1}{2}(\|u\|^2 + \|v\|^2)$. Defining $c = 1 - \frac{1}{4\mu}$ and with (3.4), we can obtain (3.1) and (3.2). The proof is complete.

We will establish the global convergence of Algorithm 1 by contradiction. Assume that there exists a positive constant $\gamma > 0$ such that

$$\|g_k\| \geq \gamma, \forall\, k \geq 1. \tag{3.5}$$

Using (3.5), we have a contradiction to get our conclusion. The following assumptions are often used to prove the convergence of the nonlinear CG methods.

Assumption 3.1 (i) The level set $\Omega = \{x \in \Re^n \mid f(x) \leq f(x_1)\}$ is bounded, where x_1 is the initial point.

(ii) In an open convex set Ω_0 that contains Ω, f has the following properties: it is a lower bound; it is differentiable; its gradient g is Lipschitz continuous, i.e. there exists a constant $L > 0$ satisfying

$$\|g(x) - g(y)\| \leq L\|x - y\|, \;\; \forall \;\; x, y \in \Omega_0. \tag{3.6}$$

Lemma 3.1 shows that the search direction possesses the sufficient descent property. Based on the relation (3.1) and Assumption 3.1 (ii), similar to Lemma 1.1 in [34], it is not difficult to get the following lemma which shows that Algorithm 1 is well defined. So we only present it as follows but omit its proof.

Lemma 3.2 *Given Assumption 3.1 hold and the sequence $\{x_k\}$ generated by Algorithm 1, then for each k, we have $f(x_k) \leq C_k \leq A_k$ Moreover, there exists α_k satisfying the nonmonotone line search conditions (2.6) and (2.7).*

Lemma 3.3 *Suppose Assumption 3.1 holds. Let the sequence $\{g_k\}$ and $\{d_k\}$ be generated by Algorithm 1. Then*

$$\alpha_k \geq \frac{1-\sigma}{L} \frac{\mid g_k^T d_k \mid}{\|d_k\|^2}, \tag{3.7}$$

$$\|y_k^m\| \leq 2L\|s_k\|, \tag{3.8}$$

and

$$\sum_{k=1}^{\infty} \frac{(g_k^T d_k)^2}{\|d_k\|^2} < +\infty \tag{3.9}$$

hold.

Proof. By (2.7) and the Lipschitz condition (3.6), we have

$$-(1-\sigma) g_k^T d_k \leq (g_{k+1} - g_k)^T d_k \leq \alpha_k L \|d_k\|^2,$$

By (3.1), we obtain (3.7). In the following, we deduce that (3.8) holds. By the mean value theorem, we have

$$
\begin{aligned}
\rho_k &= 2(f_k - f_{k+1}) + (g_{k+1} + g_k)^T s_k \\
&= (-2g(x_k + \theta s_k) + g_{k+1} + g_k)^T s_k \\
&\leq \|s_k\|[\|g_{k+1} - g(x_k + \theta s_k)\| + \|g_k - g(x_k + \theta s_k)\|] \\
&\leq \|s_k\|[L(1-\theta)\|s_k\| + L\theta\|s_k\|] \\
&= L\|s_k\|^2,
\end{aligned} \tag{3.10}
$$

where $\theta \in (0,1)$ and the last inequality follows (3.6). Thus, by the definition of y_k^m and the Lipschitz condition (3.6), we have

$$\|y_k^m\| = \|y_k + \frac{\max\{\rho_k, 0\} s_k}{\|s_k\|^2}\| \le \|y_k\| + \frac{|\rho_k| \, \|s_k\|}{\|s_k\|^2} \le 2L\|s_k\|.$$

Therefore (3.8) holds. By the definition of Q_{k+1}, $Q_1 = 1$, and the fact $0 \le \eta_{\min} \le \eta_k \le \eta_{\max} < 1$, we have

$$Q_{k+1} = 1 + \sum_{j=1}^{k} \prod_{i=0}^{j-1} \eta_{k-i} \le 1 + \sum_{j=1}^{k} \eta_{\max}^j \le \sum_{j=1}^{\infty} \eta_{\max}^j = \frac{1}{1-\eta_{\max}}. \tag{3.11}$$

By the updating relation (2.6), (3.7), and the definition of C_{k+1}, we get

$$\begin{aligned} C_{k+1} &= \frac{\eta_k Q_k C_k + f(x_k + \alpha_k d_k)}{Q_{k+1}} \\ &\le \frac{\eta_k Q_k C_k + C_k - \frac{1-\sigma}{L} \frac{(g_k^T d_k)^2}{\|d_k\|^2}}{Q_{k+1}} \\ &= C_k - \frac{\frac{1-\sigma}{L} \frac{(g_k^T d_k)^2}{\|d_k\|^2}}{Q_{k+1}}. \end{aligned} \tag{3.12}$$

Since f is bounded from below and $f_k \le C_k$ for all k, we conclude that C_k is bounded from below. From (3.11) and (3.12), it follows that

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

Therefore, (3.9) holds. This completes the proof.

Lemma 3.4 *Let Assumption 3.1 hold and the sequences $\{g_k\}$ and $\{d_k\}$ be generated by Algorithm 1. Suppose that the inequality (3.5) is true. Then we have $d_k \neq 0$ and*

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

where $u_k = \frac{d_k}{\|d_k\|}$.

Proof. The relation (3.5) and Lemma 3.1 imply that $d_k \neq 0$ holds, for otherwise $g_k = 0$, thus $u_k = \frac{d_k}{\|d_k\|}$ is reasonable. Denote

$$\delta_k = \beta_k^m \frac{\|d_k\|}{\|d_{k+1}\|}, \quad r_{k+1} = -\frac{g_{k+1}}{\|d_{k+1}\|}.$$

By (2.5), for $k \ge 1$, we have

$$u_{k+1} = r_{k+1} + \delta_k u_k.$$

Combining with $\|u_{k+1}\| = \|u_k\| = 1$, we get

$$\|r_{k+1}\| = \|u_{k+1} - \delta_k u_k\| = \|\delta_k u_{k+1} - u_k\|. \tag{3.13}$$

By $\beta_k^m \ge 0$, we get $\delta_k \ge 0$. From (3.13), it follows that from the triangular inequality

$$\begin{aligned} \|u_{k+1} - u_k\| &\le \|(1+\delta_k) u_{k+1} - (1+\delta_k) u_k\| \\ &\le \|u_{k+1} - \delta_k u_k\| + \|\delta_k u_{k+1} - u_k\| \\ &= 2\|r_{k+1}\|. \end{aligned} \tag{3.14}$$

By (1.6) and (3.9), we have

$$\sum_{k\geq 1}\frac{\|g_{k+1}\|^4}{\|d_{k+1}\|^2}=\sum_{k\geq 1}\|r_{k+1}\|^2\|g_{k+1}\|^2<\infty.$$

Combining with (3.5), we get

$$\sum_{k\geq 1}\|r_{k+1}\|^2<\infty.$$

By the above inequality and (3.14), we obtain this lemma. The proof is complete.

The following property (*) was introduced by Gilbert and Nocedal [13], which pertains to the β_k^+ formula under the sufficient descent condition. Now we show that this property (*) pertains to our method.

Property (*). Suppose that

$$0<\gamma_1\leq\|g_k\|\leq\gamma_2. \tag{3.15}$$

We say that the method has Property (*), if for all k, there exist constants $b>1$ and $\lambda>0$ such that $\mid\beta_k\mid\leq b$ and

$$\|s_k\|\leq\lambda\Rightarrow\mid\beta_k\mid\leq\frac{1}{2b}.$$

Lemma 3.5 *Let Assumption 3.1 hold and the sequences $\{g_k\}$ and $\{d_k\}$ be generated by Algorithm 1. Then the new formula β_k^m possesses property (*).*

Proof. The result of this lemma is obviously true if $\frac{g_{k+1}^T y_k^m}{d_k^T y_k^m}\leq\frac{\mu\|y_k^m\|^2}{(d_k^T y_k^m)^2}g_{k+1}^T d_k$ holds. Otherwise, from Assumption 3.1(i), there exists a constant $M_1>0$ such that

$$\|s_k\|\leq M_1. \tag{3.16}$$

By Lemma 3.2, we know that there exists α_k satisfying (2.6) and (2.7). Then, for all k, there exists a constant α_* such that $\alpha_k\geq\alpha_*$. Combining (3.16), we get

$$\alpha_*\|d_k\|\leq\alpha_k\|d_k\|=\|s_k\|\leq M_1. \tag{3.17}$$

Let $M=M_1/\alpha_*$, we have $\|d_k\|\leq M$. By the definition of β_k^m, $\|d_k\|\leq M$, (3.1), (3.2), (3.8), (3.15), and (3.16), we get

$$\begin{aligned}\mid\beta_k^m\mid &\leq \mid\frac{g_{k+1}^T y_k^m}{d_k^T y_k^m}\mid+\frac{\mu\|y_k^m\|^2}{(d_k^T y_k^m)^2}\mid g_{k+1}^T d_k\mid\\ &\leq \frac{\|g_{k+1}\|\|y_k^m\|}{c(1-\sigma)\|g_k\|^2}+\frac{\mu\|g_{k+1}\|\|d_k\|\|y_k^m\|^2}{c^2(1-\sigma)^2\|g_k\|^4}\\ &\leq \frac{2L\gamma_2\|s_k\|}{c(1-\sigma)\gamma_1^2}+\frac{4\mu\gamma_2 ML^2M_1\|s_k\|}{c^2(1-\sigma)^2\gamma_1^4}\\ &= (\frac{2cL\gamma_2\gamma_1^2(1-\sigma)+4\mu\gamma_2 ML^2M_1}{c^2(1-\sigma)^2\gamma_1^4})\|s_k\|,\end{aligned} \tag{3.18}$$

let $b=\max\{2,(\frac{2cL\gamma_2\gamma_1^2(1-\sigma)+4\mu\gamma_2 ML^2M_1}{c^2(1-\sigma)^2\gamma_1^4})M_1\}$ and $\lambda=\frac{c^2(1-\sigma)^2\gamma_1^4}{2b(2cL\gamma_2\gamma_1^2(1-\sigma)+4\mu\gamma_2 ML^2M_1)}$. From (3.18) and the definitions of b and λ, it follows that $b>1$,

$$\mid\beta_k^m\mid\leq b,$$

and

$$\begin{aligned}\mid\beta_k^m\mid &\leq (\frac{2cL\gamma_2\gamma_1^2(1-\sigma)+4\mu\gamma_2 ML^2M_1}{c^2(1-\sigma)^2\gamma_1^4})\|s_k\|\\ &\leq (\frac{2cL\gamma_2\gamma_1^2(1-\sigma)+4\mu\gamma_2 ML^2M_1}{c^2(1-\sigma)^2\gamma_1^4})\lambda\\ &= \frac{1}{2b}.\end{aligned}$$

The proof is complete.

By Lemma 3.5, similar to Lemma 3.3.2 in [8](or see [13]), it is not difficult to prove the following result. Then we only state it as follows but omit the proof.

Lemma 3.6 *Let the sequences $\{g_k\}$ and $\{d_k\}$ be generated by Algorithm 1 and the conditions in Lemma 3.5 hold. If $\beta_k^m \geq 0$ and the method has property (*), then there exists $\lambda > 0$ such that, for any $\Delta \in N$ and any index k_0, there is an index $k > k_0$ satisfying*

$$| \kappa_{k,\Delta}^{\lambda} | > \frac{\lambda}{2},$$

where $\kappa_{k,\Delta}^{\lambda} = \{i \in N : k \leq i \leq k + \Delta - 1, \|s_i\| > \lambda\}$, N denotes the set of positive integers, $| \kappa_{k,\Delta}^{\lambda} |$ denotes the numbers of elements in $\kappa_{k,\Delta}^{\lambda}$.

Based on Assumption 3.1, Lemmas 3.1-3.6, similar to Theorem 3.2 in [16], it is not difficult to get the following global convergence theorem of Algorithm 1. We only state it as follows, but omit the proof.

Theorem 3.1 *Let Assumption 3.1 hold and the sequence $\{\alpha_k, d_k, x_k, g_k\}$ be generated by Algorithm 1. Then*

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

holds.

4. Numerical Results

In this section, we test the numerical behavior of Algorithm 1. The algorithm is implemented using Fortran code in double precision arithmetic. All experiments are carried out on a PC with CPU Intel Pentium Dual E7500 2.93GHz, 2G bytes of SDRAM memory, and Red Hat Linux 9.03 operating system. Our experiments are performed on the subset of the nonlinear unconstrained problems from the CUTEr [1] collection, and the second-order derivatives of all the selected problems are available. Since we are interested in large problems, we constrain ourselves to problems in which the number of variables is at least 50. The dimension of these problems are fixed with initial points. In total, we solve 71 problems. The names and characters of these problems are listed in Table 4.1.

TABLE 4.1(Test problems and their character)

Problems	Character
ARGLINA,ARGLINB,ARGLINC,BDQRTIC,BROWNAL,BROYDN7D,BRYBND CHAINWOO,CHNROSNB,COSINE,CRAGGLVY,CURLY10,CURLY20,DIXMAANA, DIXMAANB,DIXMAANC,DIXMAAND,DIXMAANE,DIXMAANF,DIXMAANG,DIXMAANH DIXMAANI,DIXMAANJ,DIXMAANL,DIXON3DQ,DQDRTIC,DQRTIC,EDENSCH EG2,ENGVAL1,ERRINROS,EXTROSNB,FLETCBV2,FLETCHCR,FREUROTH GENHUMPS,GENROSE,INDEF,LIARWHD,MANCINO,MSQRTALS,MSQRTBLS NONCVXU2,NONDIA,NONDQUAR,PENALTY1,PENALTY2,POWELLSG POWER,QUARTC,SCHMVETT,SENSORS,SINQUAD,SPARSINE,SPARSQUR SPMSRTLS,SROSENBR,TESTQUAD,TOINTGSS,TQUARTIC,TRIDIA VARDIM,VAREIGVL,WOODS	Academic
DECONVU,FMINSRF2,FMINSURF,MOREBV,TOINTGOR,TOINTQOR	Modelling

We give the complete set of results in Appendix I. The program will be stopped when $\|g_k\|_\infty \leq \max\{10^{-6}, 10^{-12}\|g_1\|_\infty\}$ is satisfied. The parameters and the line search rules are similar to [17]: $\delta = 0.1$, $\sigma = 0.9$, $\eta_k = 0.01$, and $\mu = 0.5$. The PRP and PRP+ codes are obtained from Jorge Nocedal's Web page at

$$http://www.ece.northwestern.edu/\sim nocedal/software.html,$$

where the parameters are chosen as $\delta = 10^{-4}$, $\sigma = 10^{-1}$, the program is stopped when $\|g(x_k)\| \leq \varepsilon$ is satisfied or the inequality $\|g(x_k)\| \leq \varepsilon(1+ | f(x_k) |)$ is satisfied with $\varepsilon = 10^{-5}$. The Hager and Zhang codes are obtained from Hager's Web page at

$$http://www.math,ufl.edu/\sim hager/papers/CG.$$

Dolan and Moré [10] gave a new tool to analyze the efficiency of algorithms. They introduced the notion of a performance profile as means to evaluate and compare the performance of the set of solvers S on a test set P. Assuming that there exist n_s solvers and n_p problems, for each problem p and solver s, they defined

$t_{p,s}$ = computing time (the number of function evaluations or others) required to solve problem p by solver s.

Requiring a baseline for comparisons, they compared the performance on problem p by solver s with the best performance by any solver on this problem, based on the performance ratio

$$r_{p,s} = \frac{t_{p,s}}{\min\{t_{p,s} : s \in S\}}.$$

Suppose that a parameter $r_M \geq r_{p,s}$ for all p, s is chosen, and $r_{p,s} = r_M$ if and only if solver s does not solve problem p.

Fig.1 Performance profiles based on iterations

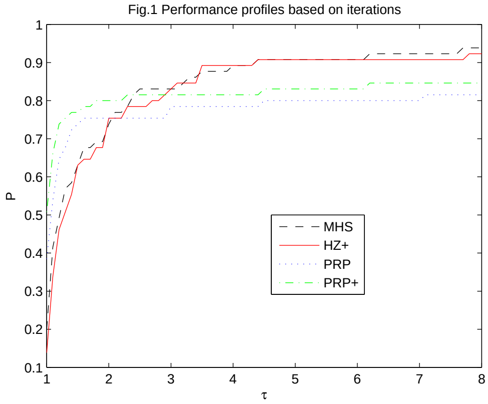


Fig.2 Performance profiles based on function

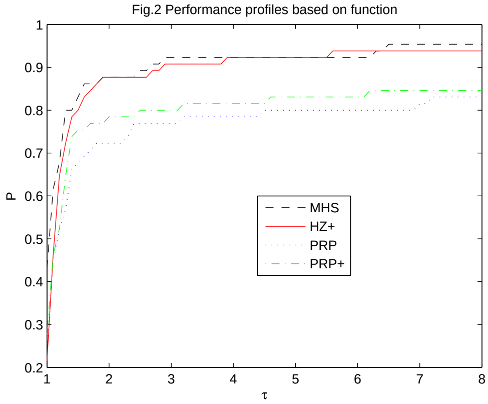
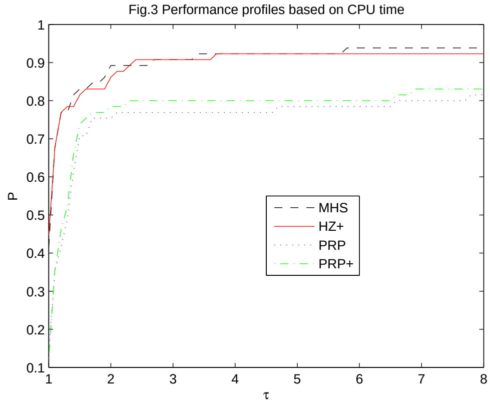


Fig.3 Performance profiles based on CPU time



The performance of solver s on any given problem might be of interest. More importantly, one would like to obtain an overall assessment of the performance of the solver. With this motivation, they defined

$$\rho_s(t) = \frac{1}{n_p} size\{p \in P : r_{p,s} \leq t\}.$$

In other words, $\rho_s(t)$ is the probability for solver $s \in S$ that a performance ratio $r_{p,s}$ is within a factor $t \in \Re$ of the best possible ration. Then function ρ_s is the (cumulative) distribution function for the performance ratio. The performance profile $\rho_s : \Re \mapsto [0, 1]$ for a solver is a nondecreasing, piecewise constant function, and continuous from the right at each breakpoint. The value of $\rho_s(1)$ is the probability that the solver would outperform the rest of the solvers.

According to the above rules, we know that one solver whose performance profile plot is on top right perform the rest of the solvers.

In these three figures, $HZ+$ denotes the algorithm in [17], MHS denotes Algorithm 1, PRP denotes the method in [23], and $PRP+$ denotes the method in [13], respectively. In Figures 1, 2, and 3, the performance refers to the iteration number, the number of function value, and the cpu time, respectively. From these three figures, all of these four methods are effective for solving most of the test problems. It is also clear that Algorithm 1 performs the best among these four algorithms. Regarding the relative performance of other three algorithms, the $HZ+$ method is better than the PRP and the $PRP+$ method, and the $PRP+$ method is comparable to the algorithm of PRP.

5. Conclusion

(i) In this paper, we propose a modified HS conjugate gradient formula based on the formula of [29] and the method of [33]. The proposed search direction possesses the sufficient descent condition without carrying out any line search. Combining with the nonmonotone line search technique [34], we obtain a nonlinear conjugate gradient algorithm. The global convergence of the algortihm is established for nonconvex functions. Numerical results show that this proposed method is competitive to $HZ+$ method, the PRP method, and the $PRP+$ method.

(ii) The given formula contains information regarding not only the gradient value but also the function value. Moreover, their quasi-Newton equation is closer to the Hessian matrix of the objective function than the normal quasi-Newton equation, which allows the method to perform well numerically.

(iii) Regarding the effectiveness of of the nonmonotone line search technique [34], we use it in the given algorithm. In fact, if the presented algorithm with the normal monotone line search technique (e.g. WWP line search) or the nonmonotone line search technique of [15] is considered, it also possesses the global convergence. In this paper, we do not analyze it anymore.

(iv) Is there good numerical performance for the presented algorithm with the normal monotone line search technique (e.g. WWP line search) or the nonmonotone line search technique of [15] ? We think this is one of the further directions to explore.

(v) Given the above discussions, there are at least three issues that warrant further improvement and research. The first point that should be considered is probably the choice of the parameters in the given CG formula, the value of the used parameters is not the only choice. Another important point is about the numerical performance, namely are there other optimality conditions and convergence conditions in the CG methods? The last one is the stop rule, there possibly exist better stop rules for the CG algorithm which improve the numerical results and convergence. All of these aspects are our further works in the future.

In summary, the proposed method indeed makes its own contribution to the literature. It also calls for more works to be done done for CG methods.

Acknowledgment. The authors would like to thank the referees' valuable comments to the idea and the language of this paper, which help improve this paper greatly.

Appendix I

Here we give the complete set of results from our tests. For each problem, in Tables 1-4 we report the number of variables (n), the number of iterations (iter), the number of function evaluations ($\#nf$), the cpu time (cpu), the value of gradient normal ($\|g(x)\|$), and the best objective function value found ($f(x)$). "\" indicates that the line search technique fails. "$-$" indicates that slope is always negative in line search."$\|$" indicates a re-entry with new function values.

TABLE 1
Test results for Algorithm 1

Problems	n	iter	$\#nf$	cpu	$\|g(x)\|$	$f(x)$
ARGLINA	200	1	3	0.03499	2.11330E-07	2.00000E+02
ARGLINB	200	5	19	0.04099	7.66930E-06	9.96250E+01
ARGLINC	200	7	18	0.03899	6.04690E-07	1.01130E+02
ARWHEAD	5000	9	24	0.08399	3.03900E-10	0
BDQRTIC	5000	282	667	0.59591	4.31400E-06	2.00060E+04
BROWNAL	200	5	11	0.019	9.93760E-07	1.47320E-09
BROYDN7D	5000	1392	2778	2.47062	7.73890E-06	2.01540E+03
BRYBND	5000	34	80	0.11398	9.38870E-06	3.72670E-11
CHAINWOO	4000	393	601	0.32195	2.89550E-06	1.00000E+00
CHNROSNB	50	229	461	0.002	6.61260E-06	1.49000E-12
COSINE	10000	10	26	0.07599	7.12210E-06	-9.99900E+03
CRAGGLVY	5000	554	1172	1.23181	8.52240E-06	1.77780E+03
CURLY10	10000	43630	69607	68.85953	9.60830E-06	-1.00320E+06
CURLY20	10000	47630	69607	136.04431	9.94700E-06	-1.00320E+06
DECONVU	61	106	213	0.004	9.66210E-06	3.14220E-07
DIXMAANA	3000	7	15	0.02	1.64690E-10	1.00000E+00
DIXMAANB	3000	8	17	0.02	3.01910E-07	1.00000E+00
DIXMAANC	3000	9	19	0.021	5.94590E-09	1.00000E+00
DIXMAAND	3000	10	21	0.021	3.12140E-06	1.00000E+00
DIXMAANE	3000	194	389	0.10498	9.99480E-06	1.00000E+00
DIXMAANF	3000	140	281	0.07999	9.54300E-06	1.00000E+00
DIXMAANG	3000	137	275	0.07899	9.79580E-06	1.00000E+00
DIXMAANH	3000	136	273	0.07799	9.69870E-06	1.00000E+00
DIXMAANI	3000	1052	2105	0.49392	9.21580E-06	1.00000E+00
DIXMAANJ	3000	143	287	0.08099	9.78490E-06	1.00000E+00
DIXMAANL	3000	116	233	0.06899	9.91190E-06	1.00000E+00
DIXON3DQ	10000	10000	20001	7.2289	2.06910E-06	2.56450E-10
DQDRTIC	5000	5	11	0.04699	6.13690E-06	2.54480E-12
DQRTIC	5000	31	63	0.02899	4.10010E-06	5.16660E-05
EDENSCH	2000	27	54	0.03299	4.68260E-06	1.20030E+04
EG2	1000	3	7	0.006	1.42150E-06	-9.98950E+02
ENGVAL1	5000	19	38	0.05999	3.68530E-06	5.54870E+03
ERRINROS	50	668	1352	0.008	8.78540E-06	3.99040E+01
EXTROSNB	1000	3007	6095	0.34895	8.81090E-06	3.19560E-06
FLETCBV2	5000	0	1	0.04899	7.99600E-08	-5.00270E-01
FLETCHCR	1000	4486	8985	0.71689	9.73440E-06	5.00120E-11
FMINSRF2	5625	286	574	0.32995	9.13150E-06	1.00000E+00
FMINSURF	5625	410	821	0.47793	9.80580E-06	1.00000E+00
FREUROTH	5000	83	165	0.17097	5.89560E-06	6.08160E+05
GENHUMPS	5000	6727	13528	10.5254	6.67590E-08	1.21150E-14
GENROSE	500	1101	2246	0.09698	7.76430E-06	1.00000E+00
INDEF	5000	–	–	–	–	–
LIARWHD	5000	22	48	0.05999	9.07610E-07	2.52840E-12
MANCINO	100	9	19	0.13898	3.31870E-06	1.90170E-17
MOREBV	5000	36	73	0.05699	9.11780E-06	8.19890E-10
MSQRTALS	1024	2413	4833	3.9344	9.83080E-06	3.56400E-08
MSQRTBLS	1024	1918	3842	3.06053	9.67840E-06	6.53590E-09
NONCVXU2	5000	8862	15495	10.55439	9.39020E-06	1.15850E+04
NONDIA	5000	8	19	0.03799	2.86400E-08	6.15440E-13
NONDQUAR	5000	770	1556	0.35195	9.27150E-06	1.76090E-05
PENALTY1	1000	33	90	0.01	5.46040E-06	9.68630E-03
PENALTY2	200	187	293	0.02699	8.94470E-06	4.71160E+13
POWELLSG	5000	204	411	0.09998	9.92850E-06	2.71230E-06
POWER	10000	332	665	0.18597	9.99380E-06	4.19040E-08
QUARTC	5000	31	63	0.02899	4.10010E-06	5.16660E-05
SCHMVETT	5000	35	63	0.12398	7.82630E-06	-1.70899E+04
SENSORS	100	29	71	0.15798	7.97470E-08	-2.09590E+03
SINQUAD	5000	44	106	0.18697	8.46720E-08	-6.75700E+06
SPARSINE	5000	16549	33100	23.31845	9.78050E-06	3.58850E-09
SPARSQUR	10000	20	41	0.12298	3.55590E-06	1.22010E-07
SPMSRTLS	4999	174	355	0.27096	9.75020E-06	1.75650E-09
SROSENBR	5000	9	19	0.02699	3.19070E-06	3.23210E-08
TESTQUAD	5000	1689	3373	0.46993	9.22200E-06	2.37400E-11
TOINTGSS	5000	4	9	0.02799	2.33390E-07	1.00020E+01
TOINTGOR	50	110	209	0.002	4.49170E-06	1.37390E+03
TOINTQOR	50	27	55	0.001	9.03850E-06	1.17550E+03
TQUARTIC	5000	21	55	0.07399	8.04370E-10	4.04390E-12
TRIDIA	5000	738	1477	0.28495	9.94720E-06	5.23100E-13
VARDIM	200	28	57	0.002	9.28820E-09	5.39190E-22
VAREIGVL	50	52	142	0.002	9.55010E-06	6.64700E-11
WOODS	4000	277	558	0.17197	8.63470E-06	4.24210E-09

TABLE 2
Test results for $HZ+$

Problems	n	iter	$\#nf$	cpu	$\|g(x)\|$	$f(x)$
ARGLINA	200	1	3	0.03599	2.11330E-07	2.00000E+02
ARGLINB	200	7	15	0.04099	7.38970E-06	9.96250E+01
ARGLINC	200	227	803	0.34195	7.14120E-06	1.01130E+02
ARWHEAD	5000	9	19	0.07899	8.39310E-07	0.00000E+00
BDQRTIC	5000	1217	2554	1.46678	8.70390E-06	2.00060E+04
BROWNAL	200	4	15	0.021	1.17900E-06	1.47310E-09
BROYDN7D	5000	1444	2885	2.52062	6.88540E-06	1.97850E+03
BRYBND	5000	31	64	0.09598	6.50550E-06	1.42920E-11
CHAINWOO	4000	272	527	0.29095	9.37220E-06	4.57280E+00
CHNROSNB	50	245	491	0.003	9.76120E-06	1.16310E-12
COSINE	10000	11	31	0.08099	2.39930E-06	-9.99900E+03
CRAGGLVY	5000	97	188	0.21397	9.78690E-06	1.68820E+03
CURLY10	10000	64092	79804	101.55256	9.37090E-06	-1.00320E+06
CURLY20	10000	100367	120307	458.68527	9.82370E-06	-1.00320E+06
DECONVU	61	102	205	0.003	9.88430E-06	3.72230E-07
DIXMAANA	3000	8	17	0.019	1.55860E-06	1.00000E+00
DIXMAANB	3000	9	19	0.021	3.55460E-07	1.00000E+00
DIXMAANC	3000	10	21	0.021	7.02360E-07	1.00000E+00
DIXMAAND	3000	11	23	0.021	3.25930E-06	1.00000E+00
DIXMAANE	3000	194	389	0.10098	9.55400E-06	1.00000E+00
DIXMAANF	3000	147	295	0.08099	9.68130E-06	1.00000E+00
DIXMAANG	3000	144	289	0.07999	9.38810E-06	1.00000E+00
DIXMAANH	3000	140	281	0.07699	9.35790E-06	1.00000E+00
DIXMAANI	3000	813	1627	0.37194	9.70570E-06	1.00000E+00
DIXMAANJ	3000	137	275	0.07599	9.61850E-06	1.00000E+00
DIXMAANL	3000	112	225	0.06499	9.91360E-06	1.00000E+00
DIXON3DQ	10000	10000	20001	6.71398	5.44620E-07	1.59400E-12
DQDRTIC	5000	7	15	0.04799	1.89570E-07	4.18260E-15
DQRTIC	5000	32	65	0.02899	3.02500E-06	2.85600E-05
EDENSCH	2000	29	56	0.03199	7.44320E-06	1.20030E+04
EG2	1000	3	7	0.006	8.13180E-06	-9.98950E+02
ENGVAL1	5000	23	45	0.06399	5.31640E-06	5.54870E+03
ERRINROS	50	1069	2136	0.011	9.20250E-06	3.99040E+01
EXTROSNB	1000	3413	6971	0.38394	7.39370E-06	3.03310E-06
FLETCBV2	5000	0	1	0.04899	7.99600E-08	-5.00270E-01
FLETCHCR	1000	6741	14004	1.10983	9.32210E-06	4.60430E-11
FMINSRF2	5625	305	613	0.33595	9.63550E-06	1.00000E+00
FMINSURF	5625	420	841	0.46693	9.56580E-06	1.00000E+00
FREUROTH	5000	65	123	0.14398	5.31530E-06	6.08160E+05
GENHUMPS	5000	6718	13563	10.78636	2.81150E-06	3.09940E-11
GENROSE	500	1267	2564	0.10498	7.13560E-06	1.00000E+00
INDEF	5000	–	–	–	–	–
LIARWHD	5000	21	48	0.05899	3.50950E-07	7.41930E-18
MANCINO	100	11	23	0.16097	9.70740E-07	2.02150E-18
MOREBV	5000	32	65	0.05199	8.97360E-06	1.01030E-09
MSQRTALS	1024	2443	4893	3.85441	9.33920E-06	2.73770E-08
MSQRTBLS	1024	1907	3820	2.98955	9.87780E-06	7.52760E-09
NONCVXU2	5000	7449	14877	7.93079	9.02800E-06	1.15850E+04
NONDIA	5000	9	29	0.04499	5.15870E-06	1.35780E-17
NONDQUAR	5000	2194	4415	0.84387	9.72040E-06	2.99280E-06
PENALTY1	1000	43	104	0.01	8.55780E-06	9.68800E-03
PENALTY2	200	181	216	0.021	9.49620E-06	4.71160E+13
POWELLSG	5000	123	250	0.06699	8.25080E-06	4.00200E-06
POWER	10000	346	693	0.17397	9.65560E-06	7.88770E-09
QUARTC	5000	32	65	0.02899	3.02500E-06	2.85600E-05
SCHMVETT	5000	35	63	0.11998	9.02750E-06	-1.49940E+04
SENSORS	100	23	55	0.13598	2.99290E-06	-2.10850E+03
SINQUAD	5000	43	106	0.18697	2.53940E-06	-6.75700E+06
SPARSINE	5000	16571	33143	22.93851	9.07100E-06	2.78210E-09
SPARSQUR	10000	20	41	0.12098	6.77340E-06	2.73360E-07
SPMSRTLS	4999	186	379	0.27996	8.24520E-06	1.95020E-09
SROSENBR	5000	12	26	0.02799	2.37120E-10	2.83550E-19
TESTQUAD	5000	1623	3247	0.41094	9.59240E-06	6.56730E-12
TOINTGSS	5000	3	7	0.02799	6.04710E-06	1.00020E+01
TOINTGOR	50	107	206	0.002	8.88940E-06	1.37390E+03
TOINTQOR	50	28	55	0.001	4.99700E-06	1.17550E+03
TQUARTIC	5000	20	61	0.07599	1.28180E-09	2.91810E-23
TRIDIA	5000	738	1477	0.26596	9.45420E-06	4.64740E-13
VARDIM	200	28	57	0.002	2.08050E-09	2.70520E-23
VAREIGVL	50	52	142	0.002	7.10790E-06	4.05080E-11
WOODS	4000	155	354	0.11798	8.41740E-06	1.39430E-08

TABLE 3
Test results for PRP

Problems	n	iter	$\#nf$	cpu	$\Vert g(x)\Vert$	$f(x)$
ARGLINA	200	1	5	0.03599	2.68600E-14	2.00000E+02
ARGLINB	200	\	\	\	\	\
ARGLINC	200	\	\	\	\	\
ARWHEAD	5000	4	15	0.07799	9.94560E-06	0.00000E+00
BDQRTIC	5000	\	\	\	\	\
BROWNAL	200	4	35	0.024	1.17460E-06	1.47300E-09
BROYDN7D	5000	6143	12485	16.22053	9.88180E-06	3.82340E+00
BRYBND	5000	37	87	0.13198	8.50590E-06	2.66050E-11
CHAINWOO	4000	‖	‖	‖	‖	‖
CHNROSNB	50	257	522	0.003	8.87240E-06	5.65700E-12
COSINE	10000	10	30	0.08099	1.08890E-06	-9.99900E+03
CRAGGLVY	5000	\	\	\	\	\
CURLY10	10000	‖	‖	‖	‖	‖
CURLY20	10000	‖	‖	‖	‖	‖
DECONVU	61	94	194	0.005	6.46460E-06	2.80250E-07
DIXMAANA	3000	8	27	0.023	1.13020E-06	1.00000E+00
DIXMAANB	3000	7	24	0.022	1.79000E-06	1.00000E+00
DIXMAANC	3000	7	25	0.023	5.43380E-07	1.00000E+00
DIXMAAND	3000	8	28	0.023	3.65780E-07	1.00000E+00
DIXMAANE	3000	190	386	0.12398	9.90230E-06	1.00000E+00
DIXMAANF	3000	136	280	0.09398	9.91130E-06	1.00000E+00
DIXMAANG	3000	125	260	0.08899	9.88610E-06	1.00000E+00
DIXMAANH	3000	135	281	0.09498	9.92060E-06	1.00000E+00
DIXMAANI	3000	704	1414	0.41394	9.96640E-06	1.00000E+00
DIXMAANJ	3000	159	327	0.10798	9.68590E-06	1.00000E+00
DIXMAANL	3000	125	261	0.08899	9.85570E-06	1.00000E+00
DIXON3DQ	10000	10000	20006	9.1886	6.18760E-08	6.56440E-13
DQDRTIC	5000	5	15	0.04899	1.25960E-08	9.86420E-16
DQRTIC	5000	47	145	0.04699	1.11750E-06	2.20930E-07
EDENSCH	2000	20	56	0.03399	6.25260E-06	1.20030E+04
EG2	1000	2	9	0.006	4.06020E-06	-9.98950E+02
ENGVAL1	5000	\	\	\	\	\
ERRINROS	50	‖	‖	‖	‖	‖
EXTROSNB	1000	53	128	0.014	8.87670E-06	3.91260E-13
FLETCBV2	5000	2402	4805	3.56646	9.97910E-06	-5.00290E-01
FLETCHCR	1000	4615	9322	0.95185	9.42350E-06	4.32530E-11
FMINSRF2	5625	270	548	0.44193	9.97120E-06	1.00000E+00
FMINSURF	5625	377	762	0.6299	9.58900E-06	1.00000E+00
FREUROTH	5000	\	\	\	\	\
GENHUMPS	5000	‖	‖	‖	‖	‖
GENROSE	500	1123	2270	0.11998	9.87490E-06	1.00000E+00
INDEF	5000	\	\	\	\	\
LIARWHD	5000	14	38	0.05499	5.20320E-06	1.00610E-11
MANCINO	100	12	29	0.19997	4.38950E-07	4.45110E-19
MOREBV	5000	36	73	0.06199	8.95260E-06	8.18030E-10
MSQRTALS	1024	2421	4847	5.64014	9.27950E-06	2.01330E-08
MSQRTBLS	1024	1893	3791	4.34934	9.77730E-06	6.41260E-09
NONCVXU2	5000	6190	12389	8.08277	9.96950E-06	1.15840E+04
NONDIA	5000	5	26	0.03899	6.58410E-08	1.43860E-17
NONDQUAR	5000	5422	10863	2.73458	9.62660E-06	8.50480E-07
PENALTY1	1000	16	72	0.009	2.39100E-06	9.68640E-03
PENALTY2	200	\	\	\	\	\
POWELLSG	5000	63	144	0.05299	2.94390E-06	8.34690E-07
POWER	10000	337	683	0.25196	9.27740E-06	3.50740E-08
QUARTC	5000	47	145	0.04699	1.11750E-06	2.20930E-07
SCHMVETT	5000	34	76	0.13098	8.74400E-06	-1.49940E+04
SENSORS	100	30	70	0.14698	3.88210E-06	-2.10180E+03
SINQUAD	5000	\	\	\	\	\
SPARSINE	5000	‖	‖	‖	‖	‖
SPARSQUR	10000	32	93	0.24896	4.28390E-06	3.54980E-08
SPMSRTLS	4999	182	370	0.35495	8.63280E-06	1.70170E-09
SROSENBR	5000	10	26	0.02899	5.47220E-06	1.48540E-10
TESTQUAD	5000	1500	3003	0.54192	8.77610E-06	2.47800E-11
TOINTGSS	5000	1	7	0.02799	7.18970E-07	1.00020E+01
TOINTGOR	50	109	220	0.002	8.69360E-06	1.37390E+03
TOINTQOR	50	26	54	0.001	5.96650E-06	1.17550E+03
TQUARTIC	5000	12	38	0.06799	3.84600E-07	4.39300E-18
TRIDIA	5000	736	1475	0.34995	9.88110E-06	4.74770E-13
VARDIM	200	12	45	0.001	3.97900E-10	9.89590E-25
VAREIGVL	50	26	54	0.001	6.86820E-06	1.77400E-11
WOODS	4000	317	654	0.24096	9.36220E-06	3.18330E-09

TABLE 4
Test results for $PRP+$

Problems	n	iter	$\#nf$	cpu	$\|g(x)\|$	$f(x)$
ARGLINA	200	1	5	0.03599	2.68600E-14	2.00000E+02
ARGLINB	200	\	\	\	\	\
ARGLINC	200	\	\	\	\	\
ARWHEAD	5000	4	15	0.07799	9.94560E-06	0.00000E+00
BDQRTIC	5000	\	\	\	\	\
BROWNAL	200	4	35	0.023	1.17460E-06	1.47300E-09
BROYDN7D	5000	6156	12507	16.25853	9.26240E-06	3.82340E+00
BRYBND	5000	37	87	0.12998	8.50590E-06	2.66050E-11
CHAINWOO	4000	268	554	0.38394	7.84150E-06	4.57280E+00
CHNROSNB	50	236	480	0.003	9.43630E-06	1.24970E-11
COSINE	10000	8	26	0.07699	2.07680E-06	-9.99900E+03
CRAGGLVY	5000	\	\	\	\	\
CURLY10	10000	‖	‖	‖	‖	‖
CURLY20	10000	‖	‖	‖	‖	‖
DECONVU	61	92	191	0.005	9.25570E-06	2.82940E-07
DIXMAANA	3000	7	20	0.021	4.60000E-07	1.00000E+00
DIXMAANB	3000	5	21	0.021	1.57950E-06	1.00000E+00
DIXMAANC	3000	7	24	0.022	4.53810E-06	1.00000E+00
DIXMAAND	3000	8	27	0.023	1.48530E-08	1.00000E+00
DIXMAANE	3000	182	370	0.11998	9.73510E-06	1.00000E+00
DIXMAANF	3000	129	265	0.08998	9.65640E-06	1.00000E+00
DIXMAANG	3000	134	277	0.09398	9.51660E-06	1.00000E+00
DIXMAANH	3000	183	375	0.12098	9.36430E-06	1.00000E+00
DIXMAANI	3000	737	1480	0.43193	9.92930E-06	1.00000E+00
DIXMAANJ	3000	152	312	0.10498	9.67660E-06	1.00000E+00
DIXMAANL	3000	121	251	0.08599	9.90490E-06	1.00000E+00
DIXON3DQ	10000	10000	20006	9.15261	6.18760E-08	6.56440E-13
DQDRTIC	5000	5	15	0.04899	1.25960E-08	9.86420E-16
DQRTIC	5000	16	64	0.03099	5.65210E-06	1.03990E-05
EDENSCH	2000	20	56	0.03499	6.97340E-06	1.20030E+04
EG2	1000	2	9	0.006	4.06020E-06	-9.98950E+02
ENGVAL1	5000	18	48	0.06499	6.45270E-06	5.54870E+03
ERRINROS	50	‖	‖	‖	‖	‖
EXTROSNB	1000	83	190	0.019	6.03050E-06	2.33400E-13
FLETCBV2	5000	2402	4805	3.54346	9.97910E-06	-5.00290E-01
FLETCHCR	1000	4487	9044	0.92686	9.53150E-06	4.40440E-11
FMINSRF2	5625	270	548	0.44393	9.97120E-06	1.00000E+00
FMINSURF	5625	377	762	0.6269	9.58900E-06	1.00000E+00
FREUROTH	5000	\	\	\	\	\
GENHUMPS	5000	9893	19992	16.88543	5.84580E-06	2.75670E-10
GENROSE	500	1121	2271	0.11998	8.12660E-06	1.00000E+00
INDEF	5000	\	\	\	\	\
LIARWHD	5000	14	37	0.05499	1.59160E-06	4.85330E-13
MANCINO	100	10	25	0.17697	2.35530E-06	1.34520E-17
MOREBV	5000	36	73	0.06099	8.95260E-06	8.18030E-10
MSQRTALS	1024	2421	4847	5.58215	9.27950E-06	2.01330E-08
MSQRTBLS	1024	1893	3791	4.38033	9.77730E-06	6.41260E-09
NONCVXU2	5000	6190	12389	8.09177	9.96950E-06	1.15840E+04
NONDIA	5000	5	26	0.03899	8.22360E-07	1.16240E-09
NONDQUAR	5000	4765	9554	2.39663	9.98140E-06	1.05410E-06
PENALTY1	1000	36	140	0.013	5.80440E-06	9.69000E-03
PENALTY2	200	\	\	\	\	\
POWELLSG	5000	75	184	0.06099	8.50510E-06	3.05250E-05
POWER	10000	334	677	0.25396	9.80170E-06	3.77930E-08
QUARTC	5000	16	64	0.03099	5.65210E-06	1.03990E-05
SCHMVETT	5000	33	74	0.12898	7.93870E-06	-1.49940E+04
SENSORS	100	29	69	0.14698	9.12830E-06	-2.10180E+03
SINQUAD	5000	\	\	\	\	\
SPARSINE	5000	‖	‖	‖	‖	‖
SPARSQUR	10000	35	100	0.26896	3.15390E-06	4.88620E-09
SPMSRTLS	4999	182	370	0.35394	8.63280E-06	1.70170E-09
SROSENBR	5000	8	23	0.02799	1.34120E-07	3.74170E-11
TESTQUAD	5000	1500	3003	0.54192	8.77610E-06	2.47800E-11
TOINTGSS	5000	1	7	0.02699	7.18970E-07	1.00020E+01
TOINTGOR	50	109	220	0.002	8.69360E-06	1.37390E+03
TOINTQOR	50	26	54	0	5.96650E-06	1.17550E+03
TQUARTIC	5000	10	36	0.06699	2.07660E-06	7.15250E-10
TRIDIA	5000	736	1475	0.35095	9.88110E-06	4.74770E-13
VARDIM	200	8	44	0.002	2.19870E-08	3.02140E-21
VAREIGVL	50	26	54	0.001	6.86820E-06	1.77400E-11
WOODS	4000	36	91	0.05199	3.26900E-07	7.89860E-14

References

[1] I. Bongartz, A. R. Conn, N. I. Gould, and P. L. Toint, CUTE: Constrained and unconstrained testing environments, ACM Trans. Math. Software, 21(1995), pp. 123-160.

[2] C. G. Broyden, J. E. Dennis, and J. J. Moré, On the local and supelinear convergence of quasi-Newton methods, Journal of the Institute of Mathematics and its Applications, 12(1973), pp. 223-246.

[3] R. Byrd and J. Nocedal, A tool for the analysis of quasi-Newton methods with application to unconstrained minimization, SIAM Journal on Numerical Analysis, 26(1989), pp. 727-739.

[4] R. Byrd, J. Nocedal, and Y. Yuan, Global convergence of a class of quasi-Newton methods on convex problems, SIAM Journal on Numerical Analysis, 24(1987), pp. 1171-1189.

[5] Y. Dai, Convergence properties of the BFGS algorithm, SIAM Journal on Optimization, 13(2003), pp. 693-701.

[6] Y. Dai and L. Z. Liao, New conjugacy conditions and related nonlinear conjugate methods, Applied Mathematics and Optimization, 43(2001), pp. 87-101.

[7] Y. Dai and Y. Yuan, A nonlinear conjugate gradient with a strong global convergence properties, SIAM Journal on Optimization, 10 (2000), pp. 177-182.

[8] Y. Dai and Y. Yuan, Nonlinear conjugate gradient Methods, Shanghai Scientific and Technical Publishers, 1998.

[9] J. E. Dennis and J. J. Moré, A characteization of superlinear convergence and its application to quasi-Newton methods, Mathematics of Computation, 28 (1974), pp. 549-560.

[10] E. D. Dolan and J. J. Moré, Benchmarking optimization software with performance profiles, Mathematical Programming, 91(2002), pp. 201-213.

[11] R. Fletcher, Practical method of optimization, Vol I: Unconstrained Optimization, 2nd edition, Wiley, New York, 1997.

[12] R. Fletcher and C. Reeves, Function minimization by conjugate gradients, The Computer Journal, 7 (1964), pp. 149-154.

[13] J. C. Gilbert and J. Nocedal, Global convergence properties of conjugate gradient methods for optimization, SIAM Journal on Optimization, 2(1992), pp. 21-42.

[14] L. Grippo and S. Lucidi, A globally convergent version of the Polak-Ribière gradient method, Mathematical Programming, 78(1997), pp. 375-391.

[15] L. Grippo, F. Lampariello, and S. Lucidi, A nonmonotone line search technique for Newton's method, SIAM journal on Numerical Analysis, 23(1986), pp. 707-716.

[16] W. W. Hager and H. Zhang, A new conjugate gradient method with guaranteed descent and an efficient line search, SIAM Journal on Optimization, 16(2005), pp. 170-192.

[17] W. W. Hager and H. Zhang, Algorithm 851: $CG_DESCENT$, A conjugate gradient method with guaranteed descent, ACM Transactions on Mathematical Software, 32(2006), pp. 113-137.

[18] M. R. Hestenes and E. Stiefel, Method of conjugate gradient for solving linear equations, Journal of Research of the National Bureau of Standards, 49(1952), pp. 409-436.

[19] D. Li and M. Fukushima, A modified BFGS method and its global convergence in nonconvex minimization, Journal of Computational and Applied Mathematics, 129(2001), pp. 15-35.

[20] D. Li and M. Fukushima, On the global convergence of the BFGS method for nonconvex unconstrained optimization problems, SIAM Journal on Optimization, 11(2001), pp. 1054-1064.

[21] X. Li and Q. Ruan, A modified PRP conjugate gradient algorithm with trust region for optimization problems, Numerical Functional Analysis and Optimization, 2(2011), pp. 496C506.

[22] G. H. Liu, J. Y. Han and D. F. Sun, Global convergence Analysis of the BFGS Algorithm with Nonmonotone linesearch, Optimization, 34(1995), pp. 147-159.

[23] E. Polak and G. Ribière, Note sur la convergence de directions conjugees, Rev. Francaise informat Recherche Opératinelle, 3(1969), pp. 35-43.

[24] Z. Wei, G. Li and L. Qi, New Quasi-Newton Methods for unconstrained optimization problems, Applied Mathematics and Computation, 175(2006), pp. 1156-1188.

[25] Z. Wei, G. Li, and L. Qi, Global convergence of the PRP conjugate gradient methods with inexact line search for nonconvex unconstrained optimization problems, Mathematics of Computation, 77(2008), pp. 2173-2193.

[26] Z. Wei, S. Yao, and L. Liu, The convergence properties of some new conjugate gradient methods, Applied Mathematics and Computation, 183(2006), pp. 1341-1350.

[27] Z. Wei, G. Yu, G. Yuan and Z. Lian, The superlinear convergence of a modified BFGS-type method for unconstrained optimization, Computational Optimization and Applications, 29(2004), pp. 315-332.

[28] G. H. Yu, Nonlinear self-scaling conjugate gradient methods for large-scale optimization problems, thesis of Doctor's Degree, Sun Yat-Sen University, 2007.

[29] G. L. Yuan, Modified nonlinear conjugate gradient methods with sufficient descent property for large-scale optimization problems, Optimization Letters, 3(2009), pp. 11-21.

[30] G. L. Yuan, A conjugate gradient method for unconstrained optimization problems, International Journal of Mathematics and Mathematical Sciences, 2009(2009), pp. 1-14.

[31] G. L. Yuan and X. W. Lu, A modified PRP conjugate gradient method, Annals of Operations Research, 166(2009), pp. 73-90.

[32] G. L. Yuan, X. W. Lu, and Z. X. Wei, A conjugate gradient method with descent direction for unconstrained optimization, Journal of Computational and Applied Mathematics, 233(2009), pp. 519-530.

[33] G. L. Yuan and Z. X. Wei, Convergence analysis of a modified BFGS method on convex minimizations, Computational Optimization and Applications, 47(2010), pp. 237-255.

[34] H. C. Zhang and W. W. Hager, A nonmonotone line search technique and its application to unconstrained optimization, SIAM Journal on Optimization, 14(2004), pp. 1043-1056.

[35] L. Zhang, W. Zhou, and D. Li, A descent modified Polak-Ribière-Polyak conjugate method and its global convergence, IMA Journal on Numerical Analysis, 26(2006), pp. 629-649.