ELSEVIER
JOURNAL OF COMPUTATIONAL AND APPLIED MATHEMATICS
Available online at www.sciencedirect.com
SciVerse ScienceDirect

(This is a sample cover image for this issue. The actual cover is not yet available at this time.)

Journal of Computational and Applied Mathematics 241 (2013) 19–29

Contents lists available at SciVerse ScienceDirect

Journal of Computational and Applied Mathematics

journal homepage: www.elsevier.com/locate/cam

A simple three-term conjugate gradient algorithm for unconstrained optimization

Neculai Andrei [1]

Research Institute for Informatics, Center for Advanced Modeling and Optimization, 8–10 Averescu Avenue, Bucharest 1, Romania

A R T I C L E   I N F O

Article history:
Received 8 March 2012
Received in revised form 28 September 2012

This paper is dedicated to the memory of Neculai Gheorghiu (1930–2009), my professor of Mathematical Analysis, Faculty of Mathematics, "*Alexandru Ioan Cuza*' University, Iaşi, Romania

Keywords:
Unconstrained optimization
Three-term conjugate gradient
Descent condition
Conjugacy condition
Numerical comparisons

A B S T R A C T

A simple three-term conjugate gradient algorithm which satisfies both the descent condition and the conjugacy condition is presented. This algorithm is a modification of the Hestenes and Stiefel algorithm (Hestenes and Stiefel, 1952) [10], or that of Hager and Zhang (Hager and Zhang, 2005) [23] in such a way that the search direction is descent and it satisfies the conjugacy condition. These properties are independent of the line search. Also, the algorithm could be considered as a modification of the memoryless BFGS quasi-Newton method. The new approximation of the minimum is obtained by the general Wolfe line search, now using a standard acceleration technique developed by Andrei (2009) [27]. For uniformly convex functions, under standard assumptions, the global convergence of the algorithm is proved. Numerical comparisons of the suggested three-term conjugate gradient algorithm versus six other three-term conjugate gradient algorithms, using a set of 750 unconstrained optimization problems, show that all these computational schemes have similar performances, the suggested one being slightly faster and more robust. The proposed three-term conjugate gradient algorithm substantially outperforms the well-known Hestenes and Stiefel conjugate gradient algorithm, as well as the more elaborate CG_DESCENT algorithm. Using five applications from the MINPACK-2 test problem collection (Averick et al., 1992) [25], with 10^6 variables, we show that the suggested three-term conjugate gradient algorithm is the top performer versus CG_DESCENT.

1. Introduction

For solving the nonlinear unconstrained optimization problem

$$\min \left\{ f(x) \;:\; x \in R^n \right\}, \tag{1.1}$$

where $f : R^n \to R$ is a continuously differentiable function, bounded from below, starting from an initial guess $x_0 \in R^n$, the nonlinear conjugate gradient method generates a sequence $\{x_k\}$ as

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

where $\alpha_k > 0$ is obtained by line search, and the directions d_k are generated as

$$d_{k+1} = -g_{k+1} + \beta_k s_k, \qquad d_0 = -g_0, \tag{1.3}$$

E-mail address: nandrei@ici.ro.

[1] N. Andrei is a member of the Academy of Romanian Scientists, Splaiul Independentei nr. 54, sector 5, Bucharest, Romania.

0377-0427/$ – see front matter © 2012 Elsevier B.V. All rights reserved.
doi:10.1016/j.cam.2012.10.002

for $k \geq 1$. In (1.3), β_k is known as the conjugate gradient parameter, $s_k = x_{k+1} - x_k$ and $g_k = \nabla f(x_k)$. The line search in conjugate gradient algorithms is often based on the general Wolfe conditions [1,2]:

$$f(x_k + \alpha_k d_k) - f(x_k) \leq \rho \alpha_k g_k^T d_k, \tag{1.4}$$

$$g_{k+1}^T d_k \geq \sigma g_k^T d_k, \tag{1.5}$$

where d_k is a descent direction and $0 < \rho \leq \sigma < 1$. However, for some conjugate gradient algorithms, a stronger version of the Wolfe line search conditions, given by (1.4) and

$$\left| g_{k+1}^T d_k \right| \leq -\sigma g_k^T d_k, \tag{1.6}$$

is needed to ensure the convergence and to enhance the stability.

Different conjugate gradient algorithms correspond to different choices for the scalar parameter β_k. The parameter β_k in (1.3) is selected so that, when applied to minimize a strongly quadratic convex function, the directions d_k and d_{k+1} are conjugate subject to the Hessian of the quadratic function. Therefore, minimizing a convex quadratic function in a subspace spanned by a set of mutually conjugate directions is equivalent to minimizing this function along each conjugate direction in turn. This is a very good idea, but the performance of these algorithms is dependent on the accuracy of the line search. However, when applied to general nonlinear functions, often the parameter β_k is computed using some other formulae which do not satisfy the conjugacy condition. For example, considering, as usual, $y_k = g_{k+1} - g_k$, then the well-known conjugate gradient methods by Fletcher and Reeves [3] ($\beta_k^{FR} = g_{k+1}^T g_{k+1}/g_k^T g_k$), Polak and Ribière [4] and Polyak [5] ($\beta_k^{PRP} = y_k^T g_{k+1}/g_k^T g_k$), Liu and Storey [6] ($\beta_k^{LS} = -y_k^T g_{k+1}/g_k^T d_k$), Dai and Yuan [7] ($\beta_k^{DY} = g_{k+1}^T g_{k+1}/y_k^T s_k$), and Dai and Liao [8] ($\beta_k^{DL} = g_{k+1}^T (y_k - t s_k)/y_k^T s_k$, where $t > 0$ is a parameter) or the conjugate descent method of Fletcher [9] ($\beta_k^{CD} = -g_{k+1}^T g_{k+1}/g_{k+1}^T d_k$) do not satisfy the conjugacy condition. By extension we call all these conjugate gradient algorithms. Observe that the Hestenes and Stiefel [10] ($\beta_k^{HS} = y_k^T g_{k+1}/y_k^T s_k$) conjugate gradient algorithm satisfies the conjugacy condition.

If f is a strongly quadratic convex function, then with an exact line search, at least in theory, all these choices for the parameter β_k in (1.3) are equivalent. For non-quadratic functions, each choice for the parameter β_k leads to very different performances of the corresponding algorithms.

In this paper we are particularly interested in *three-term conjugate gradient* methods. One of the first general three-term conjugate gradient methods was proposed by Beale [11] as

$$d_{k+1} = -g_{k+1} + \beta_k d_k + \gamma_k d_t, \tag{1.7}$$

where $\beta_k = \beta_k^{HS}$ (or β_k^{FR}, β_k^{DY} etc.),

$$\gamma_k = \begin{cases} 0, & k = t+1, \\ \dfrac{g_{k+1}^T y_t}{d_t^T y_t}, & k > t+1, \end{cases}$$

and d_t is a restart direction. McGuire and Wolfe [12] and Powell [13] made further research into the Beale three-term conjugate gradient algorithm and established an efficient restart strategy obtaining good numerical results. To make d_{k+1} satisfy the sufficient descent condition and two consecutive gradients not be far from orthogonal, the following conditions may be imposed: $-g_{k+1}^T d_{k+1} \geq \xi \left\| g_{k+1} \right\| \left\| d_{k+1} \right\|$, where ξ is a small positive constant, and the Powell–Beale restart criterion, $\left| g_k^T g_{k+1} \right| < 0.2 \left\| g_{k+1} \right\|^2$. Deng and Li [14] and Dai and Yuan [7] studied the general three-term conjugate gradient method

$$d_{k+1} = -g_{k+1} + \beta_k d_k + \gamma_k d_{t(p)}, \tag{1.8}$$

where $t(p)$ is the number of the p-th restart iteration satisfying $t(p) < k \leq t(p+1) \leq +\infty$, showing that in some mild conditions the algorithm has global convergence.

Nazareth [15] proposed a conjugate gradient algorithm using a three-term recurrence formula:

$$d_{k+1} = -y_k + \frac{y_k^T y_k}{y_k^T d_k} d_k + \frac{y_{k-1}^T y_k}{y_{k-1}^T d_{k-1}} d_{k-1}, \tag{1.9}$$

with $d_{-1} = 0$, $d_0 = 0$. If f is a convex quadratic function, then, for any steplength α_k, the search directions generated by (1.9) are conjugate subject to the Hessian of f, even without exact line search. In the same context, Zhang et al. proposed a descent modified PRP conjugate gradient algorithm with three terms [16] as

$$d_{k+1} = -g_{k+1} + \frac{g_{k+1}^T y_k}{g_k^T g_k} d_k - \frac{g_{k+1}^T d_k}{g_k^T g_k} y_k, \tag{1.10}$$

and a descent modified HS conjugate gradient algorithm with three terms [17] as

$$d_{k+1} = -g_{k+1} + \frac{g_{k+1}^T y_k}{s_k^T y_k} s_k - \frac{g_{k+1}^T s_k}{s_k^T y_k} y_k, \tag{1.11}$$

where $d_0 = -g_0$. A remarkable property of these methods is that they produce descent directions, i.e. $g_k^T d_k = -\|g_k\|$ for any $k \geq 1$. The convergence properties of (1.11) for convex optimization are given in [18].

Motivated by this nice descent property, Zhang et al. [19] introduced another three-term conjugate gradient method based on the Dai–Liao method as

$$d_{k+1} = -g_{k+1} + \frac{g_{k+1}^T(y_k - ts_k)}{y_k^T s_k} s_k - \frac{g_{k+1}^T s_k}{y_k^T s_k}(y_k - ts_k), \tag{1.12}$$

where $d_0 = -g_0$ and $t \geq 0$. Again, it is easy to see that the sufficient descent condition also holds independent of the line search used, i.e. for this method $g_k^T d_k = -\|g_k\|^2$ for all k. A specialization of this three-term conjugate gradient given by (1.12) was developed by Al-Bayati and Sharif [20], where the search direction is computed as

$$d_{k+1} = -g_{k+1} + \beta_k^{DL+} s_k - \frac{g_{k+1}^T s_k}{y_k^T s_k}(y_k - ts_k), \tag{1.13}$$

where $\beta_k^{DL+} = \max\left\{\frac{y_k^T g_{k+1}}{y_k^T s_k}, 0\right\} - t\frac{s_k^T g_{k+1}}{y_k^T s_k}$ and $t = 2\frac{\|y_k\|^2}{y_k^T s_k}$. It is easy to see that (1.13) satisfies the sufficient descent condition independent of the line search used. Cheng [21] gave another three-term conjugate gradient algorithm based on a modification of the Polak–Ribière–Polyak method:

$$d_{k+1} = -g_{k+1} + \frac{g_{k+1}^T y_k}{\|g_k\|^2}\left(I - \frac{g_{k+1} g_{k+1}^T}{\|g_{k+1}\|^2}\right) d_k, \tag{1.14}$$

proving its global convergence under appropriate line search. Another three-term conjugate gradient algorithm was given by Narushima et al. [22], where the searching direction is computed as

$$d_{k+1} = \begin{cases} -g_{k+1} & \text{if } k = 0 \text{ or } g_{k+1}^T p_k = 0, \\ -g_{k+1} + \beta_k d_k - \beta_k \frac{g_{k+1}^T d_k}{g_{k+1}^T p_k} p_k & \text{otherwise,} \end{cases} \tag{1.15}$$

where $\beta_k \in R$ is a parameter and $p_k \in R^n$ is any vector. This is a general three-term conjugate gradient method which satisfies the sufficient descent condition and for which a sufficient condition for its global convergence is proved. In the same paper, Narushima et al. [22] proposed a specific three-term conjugate gradient algorithm based on the multi-step quasi-Newton method for which the global convergence property is proved. The numerical experiments showed that the CG_DESCENT algorithm of Hager and Zhang [23] performs better than these three-term conjugate gradient algorithms. Recently, Andrei [24] suggested another three-term conjugate gradient algorithm:

$$d_{k+1} = -\frac{y_k^T s_k}{\|g_k\|^2} g_{k+1} + \frac{y_k^T g_{k+1}}{\|g_k\|^2} s_k - \frac{s_k^T g_{k+1}}{\|g_k\|^2} y_k, \tag{1.16}$$

which is a modification of the Polak–Ribière–Polyak conjugate gradient algorithm for which, independent of the line search at each iteration, both the sufficient descent condition and the conjugacy condition are satisfied. Intensive numerical experiments show that the algorithm given by (1.16) is the top performer versus PRP, DY and the three-term conjugate gradient algorithm given by (1.10).

In this paper, we present a new simple three-term conjugate gradient algorithm which is obtained by a modification of the BFGS updating scheme of the inverse approximation of the Hessian of the function f restarted as the identity matrix at every step. This computational scheme is a modification of the HS [10] or of the CG_DESCENT [23] conjugate gradient algorithms. Section 2 presents the search direction computation and its properties. It is shown that the direction satisfies both the descent and the conjugacy conditions. If the line search is exact, then the searching direction reduces to that corresponding to the Hestenes and Stiefel method. In Section 3, the three-term conjugate gradient, THREECG, endowed with an acceleration scheme and the Powell restart criterion, is presented. The convergence of the algorithm with Wolfe line search conditions is proved in Section 4. It is shown that for uniformly convex functions the directions generated by the algorithm are bounded above. In Section 5, some numerical results and comparisons with some other conjugate gradient algorithms are presented. It is shown that the performances of this conjugate gradient algorithm with three terms are similar to those of some other three-term conjugate gradient algorithms, in some cases THREECG being slightly faster. Compared with the HS and CG_DESCENT conjugate gradient algorithms, THREECG appears to be way faster and more robust. Finally, using five applications from the MINPACK-2 test problem collection [25], with 10^6 variables, we show that THREECG is significantly better than CG_DESCENT.

2. A simple three-term conjugate gradient algorithm

In this section, we describe our three-term conjugate gradient algorithm for which, independent of the line search, at every step both the *descent condition* and the *conjugacy condition* are satisfied. The algorithm is given by (1.2), where the direction d_{k+1} is computed as

$$d_{k+1} = -g_{k+1} - \delta_k s_k - \eta_k y_k, \tag{2.1}$$

where

$$\delta_k = \left(1 + \frac{\|y_k\|^2}{y_k^T s_k}\right) \frac{s_k^T g_{k+1}}{y_k^T s_k} - \frac{y_k^T g_{k+1}}{y_k^T s_k}, \tag{2.2}$$

$$\eta_k = \frac{s_k^T g_{k+1}}{y_k^T s_k}. \tag{2.3}$$

Observe that the direction d_{k+1} from (2.1) can be written as

$$d_{k+1} = -Q_k g_{k+1}, \tag{2.4}$$

where the matrix Q_k is given by

$$Q_k = I - \frac{s_k y_k^T - y_k s_k^T}{y_k^T s_k} + \left(1 + \frac{\|y_k\|^2}{y_k^T s_k}\right) \frac{s_k s_k^T}{y_k^T s_k}. \tag{2.5}$$

As we know, the BFGS updating of the inverse approximation of the Hessian of function f is

$$H_{k+1} = H_k - \frac{s_k y_k^T H_k + H_k y_k s_k^T}{y_k^T s_k} + \left(1 + \frac{y_k^T H_k y_k}{y_k^T s_k}\right) \frac{s_k s_k^T}{y_k^T s_k}. \tag{2.6}$$

Obviously, the matrix Q_k in (2.5) is a modification of the BFGS updating (2.6) in the sense that it is restarted with the identity matrix at every step ($H_k = I$), and more importantly the sign in front of $y_k s_k^T$ in the second term of (2.5) is modified to get the descent property, as is proved in the following proposition.

Proposition 2.1. *Suppose that the line search satisfies the Wolfe conditions (1.4) and (1.5). Then d_{k+1} given by (2.1) and (2.2)–(2.3) is a descent direction.*

Proof. Since the line search satisfies the Wolfe conditions, it follows that $y_k^T s_k > 0$. Now, by direct computation, we have

$$g_{k+1}^T d_{k+1} = -\|g_{k+1}\|^2 - \left(1 + \frac{\|y_k\|^2}{y_k^T s_k}\right) \frac{(s_k^T g_{k+1})^2}{y_k^T s_k} \le 0. \quad \square$$

As we know, Day and Liao [8] extended the classical conjugate condition $y_k^T d_{k+1} = 0$, suggesting the following one: $y_k^T d_{k+1} = -t(s_k^T g_{k+1})$, where $t \ge 0$ is a scalar. The proposition below proves that the direction d_{k+1} given by (2.1)–(2.3) satisfies the Dai–Liao conjugacy condition.

Proposition 2.2. *Suppose that the line search satisfies the Wolfe conditions (1.4) and (1.5). Then d_{k+1} given by (2.1)–(2.3) satisfies the conjugacy condition $y_k^T d_{k+1} = -t_k(s_k^T g_{k+1})$, where $t_k > 0$ for all k.*

Proof. By direct computation, we get

$$y_k^T d_{k+1} = -\left(1 + 2\frac{\|y_k\|^2}{y_k^T s_k}\right)(s_k^T g_{k+1}) \equiv -t_k(s_k^T g_{k+1}), \tag{2.7}$$

where $t_k = \left(1 + 2\frac{\|y_k\|^2}{y_k^T s_k}\right) > 0$, since $y_k^T s_k > 0$. $\square$

Observe that, if f is strongly convex or the line search satisfies the Wolfe conditions (1.4) and (1.5), then $y_k^T s_k > 0$, and therefore the computational scheme above yields descent. Besides, the direction d_{k+1} satisfies the Dai–Liao conjugacy condition (2.7), where $t_k > 0$ at every iteration. On the other hand, if the line search is exact, i.e. $s_k^T g_{k+1} = 0$, then (2.1) reduces to the Hestenes–Stiefel method.

3. The THREECG algorithm

In this section, we present the algorithm THREECG, in which the search direction is given by (2.1) and the parameters δ_k and η_k are computed as in (2.2) and (2.3), respectively. We know that in conjugate gradient algorithms the search directions tend to be poorly scaled, and as a consequence the line search must perform more function evaluations in order to obtain a suitable steplength α_k. Besides, in conjugate gradient methods the steplengths computed by means of the Wolfe line search (1.4) and (1.5) may differ from 1 in a very unpredictable manner [26]. They can be larger or smaller than 1 depending on how the problem is scaled. This is in very sharp contrast to the Newton and quasi-Newton methods, including the limited memory quasi-Newton methods, which accept the unit steplength most of the time during the iterations, and therefore usually they require only a few function evaluations per search direction. In this section, we take advantage of this behavior of conjugate gradient algorithms and consider an acceleration scheme we have presented in [27] (see also [28]).

Basically, the acceleration scheme modifies the steplength α_k in a multiplicative manner to improve the reduction of the function values during the iterations. As in [27], in the accelerated algorithm, instead of (1.2), the new estimation of the minimum point is computed as

$$x_{k+1} = x_k + \xi_k \alpha_k d_k, \tag{3.1}$$

where

$$\xi_k = -\frac{\bar{a}_k}{\bar{b}_k}, \tag{3.2}$$

$\bar{a}_k = \alpha_k g_k^T d_k$, $\bar{b}_k = -\alpha_k (g_k - g_z)^T d_k$, $g_z = \nabla f(z)$ and $z = x_k + \alpha_k d_k$. Hence, if $\bar{b}_k > 0$, then the new estimation of the solution is computed as $x_{k+1} = x_k + \xi_k \alpha_k d_k$; otherwise, $x_{k+1} = x_k + \alpha_k d_k$. Observe that $\bar{b}_k = \alpha_k (g_z - g_k)^T d_k = \alpha_k (d_k^T \nabla^2 f(\bar{x}_k) d_k)$, where $\bar{x}_k$ is a point on the line segment connecting x_k and z. Since $\alpha_k > 0$, it follows that for convex functions $\bar{b}_k \geq 0$. For uniformly convex functions, we can prove the linear convergence of the acceleration scheme [27].

Therefore, taking into consideration this acceleration scheme and using the definitions of g_k, s_k and y_k, the following conjugate gradient algorithm can be presented.

The THREECG algorithm

Step 1.	Select a starting point $x_0 \in \operatorname{dom} f$ and compute $f_0 = f(x_0)$ and $g_0 = \nabla f(x_0)$. Select some positive values for ρ and σ. Set $d_0 = -g_0$ and $k = 0$.
Step 2.	Test a criterion for stopping the iterations. If the test is satisfied, then stop; otherwise continue with step 3.
Step 3.	Determine the steplength α_k by using the Wolfe line search conditions (1.4)–(1.5).
Step 4.	Compute $z = x_k + \alpha_k d_k$, $g_z = \nabla f(z)$ and $y_k = g_k - g_z$.
Step 5.	Compute $\bar{a}_k = \alpha_k g_k^T d_k$ and $\bar{b}_k = -\alpha_k y_k^T d_k$.
Step 6.	Acceleration scheme. If $\bar{b}_k > 0$, then compute $\xi_k = -\bar{a}_k / \bar{b}_k$, and update the variables as $x_{k+1} = x_k + \xi_k \alpha_k d_k$; otherwise, update the variables as $x_{k+1} = x_k + \alpha_k d_k$. Compute f_{k+1} and g_{k+1}. Compute $y_k = g_{k+1} - g_k$ and $s_k = x_{k+1} - x_k$.
Step 7.	Determine δ_k and η_k as in (2.2) and (2.3), respectively.
Step 8.	Compute the search direction as $d_{k+1} = -g_{k+1} - \delta_k s_k - \eta_k y_k$.
Step 9.	Powell restart criterion. If $\left| g_{k+1}^T g_k \right| > 0.2 \left\| g_{k+1} \right\|^2$, then set $d_{k+1} = -g_{k+1}$.
Step 10.	Consider $k = k + 1$ and go to step 2. $\square$

If f is bounded along the direction d_k, then there exists a stepsize α_k satisfying the Wolfe line search conditions (1.4) and (1.5). In our algorithm, when the Powell restart condition is satisfied, we restart the algorithm with the negative gradient $-g_{k+1}$. More sophisticated reasons for restarting the algorithms have been proposed in the literature [29], but we are interested in the performance of a conjugate gradient algorithm that uses this restart criterion associated to a direction satisfying both the descent and the conjugacy conditions. Under reasonable assumptions, the Wolfe line search conditions and the Powell restart criterion are sufficient to prove the global convergence of the algorithm. The first trial of the steplength crucially affects the practical behavior of the algorithm. At every iteration $k \geq 1$ the starting guess for the step α_k in the line search is computed as $\alpha_{k-1} \left\| d_{k-1} \right\| / \left\| d_k \right\|$. This selection was used for the first time by Shanno and Phua in CONMIN [30] and in SCALCG by Andrei [31].

4. Convergence analysis

Assume the following.

(i) *The level set $S = \{x \in R^n : f(x) \leq f(x_0)\}$ is bounded, i.e. there exists positive constant $B > 0$ such that, for all $x \in S$, $\|x\| \leq B$.*

(ii) *In a neighborhood N of S the function f is continuously differentiable and its gradient is Lipschitz continuous, i.e. there exists a constant $L > 0$ such that $\|\nabla f(x) - \nabla f(y)\| \leq L \|x - y\|$, for all $x, y \in N$.*

Under these assumptions on f, there exists a constant $\Gamma \geq 0$ such that $\|\nabla f(x)\| \leq \Gamma$, for all $x \in S$. Observe that the assumption that the function f is bounded below is weaker than the usual assumption that the level set is bounded.

Although the search directions generated by (2.1)–(2.3) are always descent directions, to ensure convergence of the algorithm we need to constrain the choice of the steplength α_k. The following proposition shows that the Wolfe line search always gives a lower bound for the steplength α_k.

Proposition 4.1. *Suppose that d_k is a descent direction and that the gradient ∇f satisfies the Lipschitz condition*

$$\|\nabla f(x) - \nabla f(x_k)\| \leq L\|x - x_k\|$$

for all x on the line segment connecting x_k and x_{k+1}, where L is a positive constant. If the line search satisfies the Wolfe conditions (1.4) *and* (1.5), *then*

$$\alpha_k \geq \frac{(1-\sigma)\left|g_k^T d_k\right|}{L\|d_k\|^2}. \tag{4.1}$$

Proof. Subtracting $g_k^T d_k$ from both sides of (1.5), and using Lipschitz continuity, we get

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

Since d_k is a descent direction and $\sigma < 1$, (4.1) follows immediately. □

To prove the global convergence of nonlinear conjugate gradient algorithms, often the Zoutendijk condition is used. Under the Wolfe line search this condition was given by Wolfe [1,2] and Zoutendijk [32]. The following proposition proves that, in the above three-term conjugate gradient method, the Zoutendijk condition holds under the general Wolfe line search (1.4) and (1.5).

Proposition 4.2. *Suppose that assumptions* (i) *and* (ii) *hold. Consider the algorithm* (1.2) *and* (2.1)–(2.3), *where d_k is a descent direction and α_k is computed by the general Wolfe line search* (1.4) *and* (1.5). *Then*

$$\sum_{k=0}^{\infty} \frac{(g_k^T d_k)^2}{\|d_k\|^2} < +\infty. \tag{4.2}$$

Proof. From (1.4) and Proposition 4.1, we get

$$f_k - f_{k+1} \geq -\rho\alpha_k g_k^T d_k \geq \rho\frac{(1-\sigma)(g_k^T d_k)^2}{L\|d_k\|^2}.$$

Therefore, from assumption (i), we get the Zoutendijk condition (4.2). □

In conjugate gradient algorithms, the iteration can fail, in the sense that $\|g_k\| \geq \gamma > 0$ for all k, only if $\|d_k\| \to \infty$ sufficiently rapidly [13]. More exactly, the sequence of gradient norms $\|g_k\|$ can be bounded away from zero only if $\sum_{k\geq 0} 1/\|d_k\| < \infty$. For any conjugate gradient method with strong Wolfe line search (1.4) and (1.6), the following general result holds [26].

Proposition 4.3. *Suppose that assumptions* (i) *and* (ii) *hold, and consider any conjugate gradient algorithm* (1.2), *where d_k is a descent direction and α_k is obtained by the strong Wolfe line search* (1.4) *and* (1.6). *If*

$$\sum_{k\geq 1} \frac{1}{\|d_k\|^2} = \infty, \tag{4.3}$$

then

$$\lim_{k\to\infty}\inf \|g_k\| = 0. \tag{4.4}$$

For *uniformly convex functions*, we can prove that the norm of the direction d_{k+1} generated by (2.1)–(2.3) is bounded above. Therefore, by Proposition 4.3, we can prove the following result.

Theorem 4.1. *Suppose that assumptions* (i) *and* (ii) *hold, and consider the algorithm* (1.2) *and* (2.1)–(2.3), *where d_k is a descent direction and α_k is computed by the strong Wolfe line search* (1.4) *and* (1.6). *Suppose that f is a uniformly convex function on S, i.e. there exists a constant $\mu > 0$ such that*

$$(\nabla f(x) - \nabla f(y))^T(x - y) \geq \mu\|x - y\|^2 \tag{4.5}$$

for all $x, y \in N$*; then*

$$\lim_{k\to\infty} \|g_k\| = 0. \tag{4.6}$$

Proof. From Lipschitz continuity, we have $\|y_k\| \le L\|s_k\|$. On the other hand, from uniform convexity, $y_k^T s_k \ge \mu \|s_k\|^2$. Using the Cauchy inequality, assumptions (i) and (ii) and the above inequalities, we have

$$\begin{aligned} |\delta_k| &\le \frac{|s_k^T g_{k+1}|}{|y_k^T s_k|} + \frac{\|y_k\|^2 |s_k^T g_{k+1}|}{|y_k^T s_k|^2} + \frac{|y_k^T g_{k+1}|}{|y_k^T s_k|} \le \frac{\Gamma}{\mu\|s_k\|} + \frac{L^2\Gamma}{\mu^2\|s_k\|} + \frac{L\Gamma}{\mu\|s_k\|} \\ &= \frac{\Gamma}{\mu}\left(1 + L + \frac{L^2}{\mu}\right)\frac{1}{\|s_k\|}. \end{aligned} \tag{4.7}$$

At the same time,

$$|\eta_k| = \frac{|s_k^T g_{k+1}|}{|y_k^T s_k|} \le \frac{\|s_k\|\,\|g_{k+1}\|}{\mu\|s_k\|^2} \le \frac{\Gamma}{\mu\|s_k\|}. \tag{4.8}$$

Therefore, using (4.7) and (4.8) in (2.1), we get

$$\|d_{k+1}\| \le \|g_{k+1}\| + |\delta_k|\,\|s_k\| + |\eta_k|\,\|y_k\| \le \Gamma + \frac{\Gamma}{\mu}\left(2 + L + \frac{L^2}{\mu}\right), \tag{4.9}$$

showing that (4.3) is true. By Proposition 4.3, it follows that (4.4) is true, which for uniformly convex functions is equivalent to (4.6). □

5. Numerical results and comparisons

In this section, we report some numerical results obtained with an implementation of the THREECG algorithm. The code is written in Fortran and compiled with f77 (default compiler settings) on a Workstation Intel Pentium 4 with 1.8 GHz. THREECG, like all codes of algorithms considered in this numerical study, uses loop unrolling to a depth of 5. We selected 75 large-scale unconstrained optimization test functions in the generalized or extended form we presented in [33]. For each test function, we undertook ten numerical experiments with the number of variables increasing as $n = 1000, 2000, \ldots, 10000$. The algorithm implements the Wolfe line search conditions with $\rho = 0.0001$, $\sigma = 0.8$ and the same stopping criterion $\|g_k\|_\infty \le 10^{-6}$, where $\|\cdot\|_\infty$ is the maximum absolute component of a vector. In all the algorithms we considered in this numerical study the maximum number of iterations is limited to 10,000.

The comparisons of algorithms are given in the following context. Let f_i^{ALG1} and f_i^{ALG2} be the optimal values found by ALG1 and ALG2, for problem $i = 1, \ldots, 750$, respectively. We say that, in the particular problem i, the performance of ALG1 was better than the performance of ALG2 if

$$\left|f_i^{ALG1} - f_i^{ALG2}\right| < 10^{-3} \tag{5.1}$$

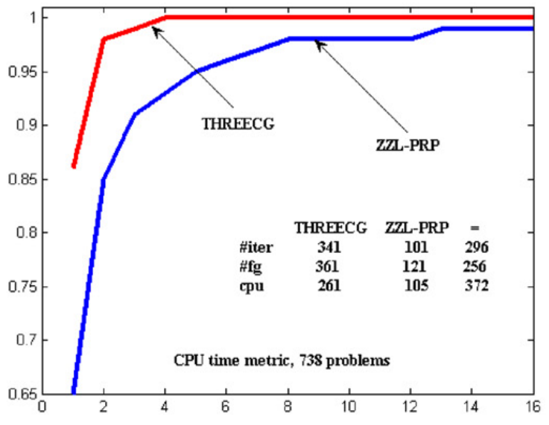
and the number of iterations (#iter), or the number of function-gradient evaluations (#fg), or the CPU time of ALG1 was less than the number of iterations, or the number of function-gradient evaluations, or the CPU time corresponding to ALG2, respectively.

In the first set of numerical experiments we compare THREECG versus ZZL-PRP (1.10) [16], ZZL-HS (1.11) [17], ZXW (1.12) [19], ABS (1.13) [20], CHENG (1.14) [21] and PRP-DC (1.16) [24]. Fig. 1 shows the Dolan and Moré CPU performance profile of THREECG versus all these three-term conjugate gradient algorithms.

In a performance profile plot, the top curve corresponds to the method that solved the most problems in a time that was within a given factor of the best time. The percentage of the test problems for which a method is the fastest is given on the left axis of the plot. The right side of the plot gives the percentage of the test problems that were successfully solved by these algorithms, respectively. The right side is a measure of the robustness of an algorithm.

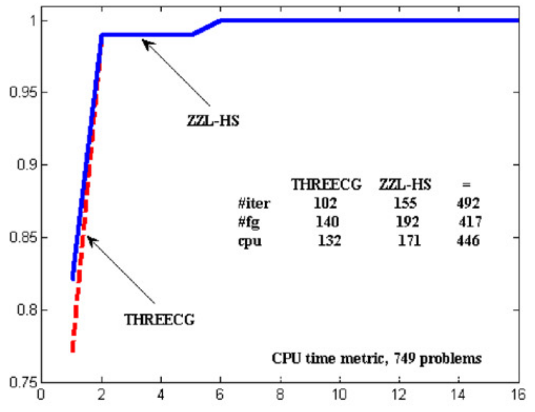
When comparing THREECG with the ZZL-PRP three-term conjugate gradient algorithm subject to CPU time metric, for example, we see that THREECG is the top performer. The three-term accelerated conjugate gradient algorithm THREECG is more successful and more robust than the three-term ZZL-PRP conjugate gradient algorithm. Comparing THREECG (see Fig. 1), subject to the number of iterations, we see that THREECG was better in 341 problems (i.e. it achieved the minimum number of iterations in 341 problems). ZZL-PRP was better in 101 problems, and they had the same number of iterations in 296 problems. Out of 750 problems, only for 738 problems does the criterion (5.1) hold. Therefore, in comparison with ZZL-PRP, THREECG appears to generate the best search direction and the best steplength, on average.

However, things are not what they seem to be. From Fig. 1, we see that for solving large-scale unconstrained optimization problems the three-term conjugate gradient algorithms have similar performances. A close analysis of three-term conjugate gradient algorithms shows that the search direction d_{k+1} is obtained as a linear combination of g_{k+1}, d_k and y_k, where



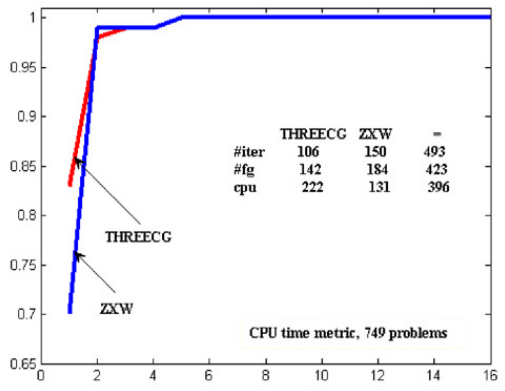
	THREECG	ZZL-PRP	=
#iter	341	101	296
#fg	361	121	256
cpu	261	105	372

THREECG versus ZZL-PRP



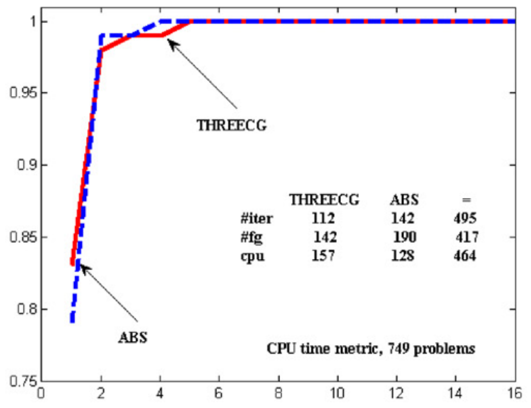
	THREECG	ZZL-HS	=
#iter	102	155	492
#fg	140	192	417
cpu	132	171	446

THREECG versus ZZL-HS



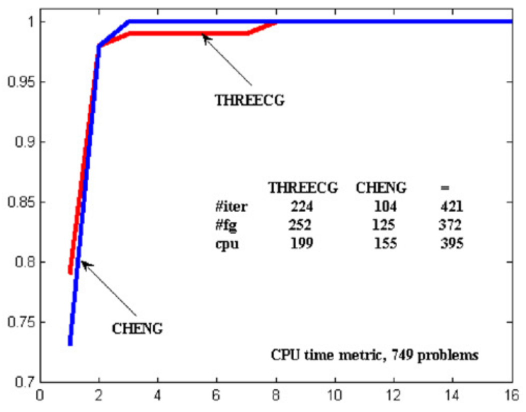
	THREECG	ZXW	=
#iter	106	150	493
#fg	142	184	423
cpu	222	131	396

THREECG versus ZXW



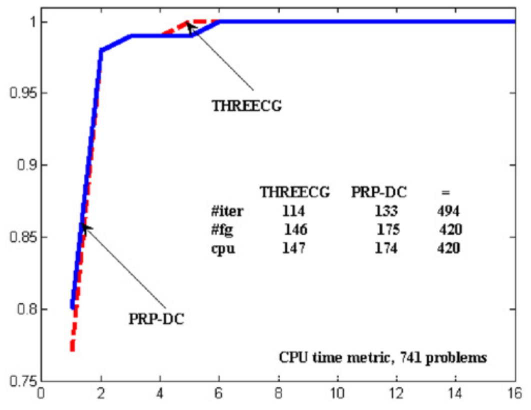
	THREECG	ABS	=
#iter	112	142	495
#fg	142	190	417
cpu	157	128	464

THREECG versus ABS



	THREECG	CHENG	=
#iter	224	104	421
#fg	252	125	372
cpu	199	155	395

THREECG versus CHENG



	THREECG	PRP-DC	=
#iter	114	133	494
#fg	146	175	420
cpu	147	174	420

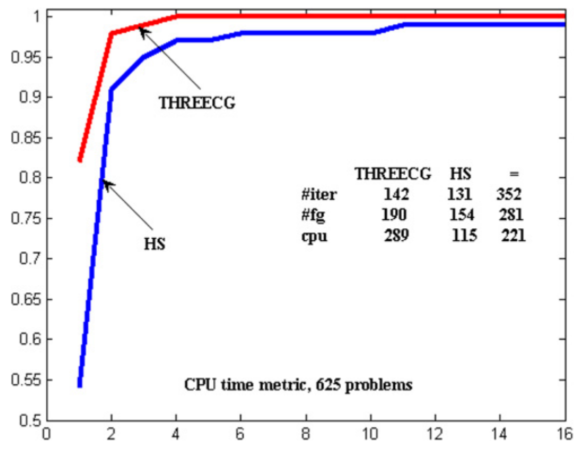
THREECG versus PRP-DC

Fig. 1. THREECG versus ZZL-PRP, ZZL-HS, ZXW, ABS, CHENG and PRP-DC.

the coefficients in these linear combinations are computed using the same elements as $\|y_k\|^2$, $\|g_k\|^2$, $\|g_{k+1}\|^2$, $s_k^T g_{k+1}$ and $y_k^T g_{k+1}$ in similar computational formulae, in order to satisfy the descent property. Therefore their numerical performances are similar.

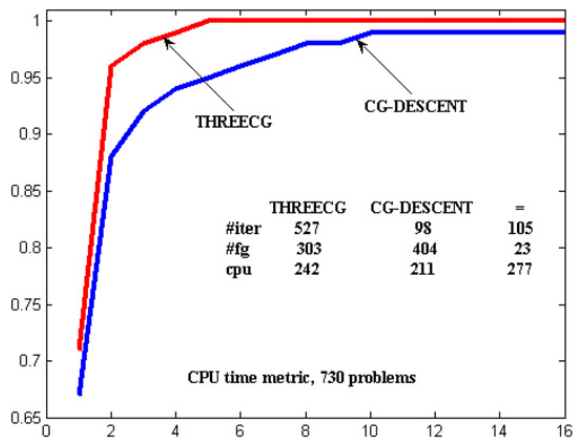
Having in view that, when the line search is exact, (2.1) reduces to the Hestenes–Stiefel method, in the second set of numerical experiments we compare our three-term conjugate gradient algorithm THREECG versus the HS method. Fig. 2 shows the Dolan and Moré CPU performance profile corresponding to these algorithms. Clearly, THREECG is the top performer in this case.

The HS method has the property that the conjugacy condition $y_k^T d_{k+1} = 0$ always holds, independent of the line search. Also, the HS algorithm is not susceptible to jamming, which is a very important property. Using an exact line search, $\beta_k^{HS} = \beta_k^{PRP}$. Therefore, the convergence properties of the HS algorithm should be similar to the convergence properties of the PRP algorithm. But, for general nonlinear functions the convergence of the PRP method is uncertain. The classical Powell



	THREECG	HS	=
#iter	142	131	352
#fg	190	154	281
cpu	289	115	221

Fig. 2. THREECG versus HS.



	THREECG	CG-DESCENT	=
#iter	527	98	105
#fg	303	404	23
cpu	242	211	277

Fig. 3. THREECG versus CG_DESCENT.

example shows that, when the function is not strongly convex, the PRP algorithm may not converge, even with an exact line search [13]. Therefore, the HS algorithm with an exact line search may not converge for general nonlinear functions. However, although HS (and PRP) may not converge in general, they often perform better than some other conjugate gradient algorithms like FR, DY and CD.

On the other hand, observe that the THREECG algorithm is a modification of the HS algorithm. Indeed, the direction given by (2.1)–(2.3) can be written as

$$d_{k+1} = -g_{k+1} + \frac{y_k^T g_{k+1}}{y_k^T s_k} s_k - \left[\left(1 + \frac{\|y_k\|^2}{y_k^T s_k}\right)\frac{1}{y_k^T s_k} s_k - \frac{1}{y_k^T s_k} y_k\right]\left(s_k^T g_{k+1}\right). \tag{5.2}$$

However, $s_k^T g_{k+1}$ tends to zero during the iterations. Apparently, this contribution to the HS direction, which tends to zero during the iterations, determines a better direction for minimizing the function f. It is worth saying that, with a standard Wolfe line search, $y_k^T s_k > 0$ when $g_k \neq 0$. Hence, when the iterates jam, y_k becomes tiny, while $\|g_k\|$ is bounded away from zero. Consequently, when the iterates jam, the expression $(\|y_k\|^2\ (s_k^T g_{k+1}))/(y_k^T s_k)^2$ in the above formulation of the direction becomes negligible. This is the reason why THREECG is faster and more robust than HS.

In the third set of numerical experiments, we compare THREECG versus CG_DESCENT with Wolfe line search [23]. Fig. 3 presents the Dolan and Moré CPU performance profile corresponding to these algorithms.

CG_DESCENT was devised in order to ensure sufficient descent, independent of the accuracy of the line search. In CG_DESCENT the search direction $d_{k+1} = -g_{k+1} + \beta_k s_k$, where

$$\beta_k^{HZ} = \left(y_k - 2\frac{\|y_k\|^2}{y_k^T s_k} s_k\right)^T \frac{g_{k+1}}{y_k^T s_k}, \tag{5.3}$$

satisfies the sufficient descent condition $g_k^T d_k \leq -(7/8)\ \|g_k\|^2$. If the function f is a quadratic and the line search is exact, then CG_DESCENT reduces to HS. In fact, CG_DESCENT is a modification of the HS algorithm. However, in CG_DESCENT the search directions do not satisfy the conjugacy condition. Again, when iterates jam, the expression $(\|y_k\|^2\ (s_k^T g_{k+1}))/(y_k^T s_k)^2$ in the above formulation of β_k^{HZ} becomes negligible. This modification of the HS scheme makes CG_DESCENT perform better than HS [23].

Table 1
Applications from the MINPACK-2 collection.

A1	*Elastic–plastic torsion* [34, pp. 41–55], $c = 5$
A2	*Pressure distribution in a journal bearing* [35], $b = 10$, $\varepsilon = 0.1$
A3	*Optimal design with composite materials* [36], $\lambda = 0.008$
A4	*Steady-state combustion* [37, pp. 292–299], [38], $\lambda = 5$
A5	*Minimal surfaces with enneper conditions* [39, pp. 80–85]

Table 2
Performance of THREECG versus CG_DESCENT 1,000,000 variables. CPU seconds.

	THREECG			CG_DESCENT		
	#iter	#fg	CPU	#iter	#fg	CPU
A1	1,111	2,253	**306.04**	1,145	2,291	436.05
A2	2,837	5,702	**979.27**	3,368	6,737	1571.53
A3	4,507	9,104	**1904.79**	4,841	9,684	2904.12
A4	1,413	2,864	**1128.70**	1,806	3,613	2093.79
A5	1,333	2,689	**546.20**	1,226	2,453	713.89
TOTAL	11,201	22,612	**4865.00**	12,386	24,778	7719.38

On the other hand, in the THREECG algorithm the search direction is a descent one, and it satisfies the conjugacy condition $y_k^T d_{k+1} = -t_k(s_k^T g_{k+1})$, where t_k is a positive parameter which is changed at every step independent of the line search. Besides, excepting the omission of the factor 2 in the term $((\|y_k\|^2 (s_k^T g_{k+1}))/(y_k^T s_k)^2)s_k$ from the direction d_{k+1} in (5.2), THREECG represents a modification of CG_DESCENT. Again, the best performance, relative to the CPU time metric, was obtained by THREECG, the top curve in Fig. 3.

6. Solving MINPACK-2 applications

We now present comparisons between THREECG and CG_DESCENT conjugate gradient algorithms for solving some applications from the MINPACK-2 test problem collection [25]. In Table 1, we present these applications, as well as the values of their parameters. The infinite-dimensional version of these problems is transformed into a finite element approximation by triangulation. Thus a finite-dimensional minimization problem is obtained whose variables are the values of the piecewise linear function at the vertices of the triangulation. The discretization steps are $nx = 1000$ and $ny = 1000$, thus obtaining minimization problems with 1,000,000 variables.

A comparison between THREECG (Powell restart criterion, $\|\nabla f(x_k)\|_\infty \le 10^{-6}$, $\rho = 0.0001$, $\sigma = 0.8$) and CG_DESCENT (Wolfe line search, default settings, $\|\nabla f(x_k)\|_\infty \le 10^{-6}$) for solving these applications is given in Table 2.

Form Table 2, we see that, subject to the CPU time metric, the THREECG algorithm is the top performer again, and the difference is significant, about **2854.38** s for solving all these five applications.

The THREECG and CG_DESCENT algorithms (and codes) are different in many respects. Since both of them use the Wolfe line search (however, implemented in different manners), these codes mainly differ in their choice of the search direction. THREECG appears to generate a better search direction, on average. The direction d_{k+1} given by (2.1)–(2.3) used in THREECG is more elaborate: it satisfies both the descent condition and the conjugacy condition in a restart environment. The three-term conjugate gradient THREECG computational scheme proved to be more efficient and more robust in numerical experiments and applications.

7. Conclusions

Three-term conjugate gradient methods represent an interesting computational innovation which produce efficient conjugate gradient algorithms. Plenty of three-term conjugate gradient algorithms can be generated, and we can suggest the following *project* in this respect. The search direction of these algorithms belonging to this project is computed like

$$d_{k+1} = -g_{k+1} - a_k s_k - b_k y_k,$$

as modifications of the classical conjugate gradient algorithms FR, PRP, LS, DY, DL, CD, etc., where the scalar parameters a_k and b_k are determined in such a way that the descent condition or/and the conjugacy condition is to be satisfied. Of course some other conditions on d_{k+1} may be introduced. Using this idea a lot of three-term conjugate gradient algorithms can be developed.

In this paper, a new three-term conjugate gradient algorithm, as a modification of HS or as a modification of CG_DESCENT based on a modification of the memoryless BFGS quasi-Newton updating, for which both the descent condition and the conjugacy condition are satisfied, has been presented. The convergence of this algorithm, for uniformly convex functions, was proved using classical assumptions. Intensive numerical experiments using 750 unconstrained optimization test problems of different characteristics proved that the suggested algorithm is the top performer in its class of three-term conjugate gradient algorithms, and is faster and more robust versus HS and CG_DESCENT algorithms. Using five applications from the MINPACK-2 test problem collection, with 10^6 variables, we showed that THREECG is obviously better than CG_DESCENT. From the above numerical experiments with 750 test problems and five real applications, we have computational evidence that the three-term conjugate gradient algorithm THREECG represents a distinct new computational scheme that is easy to understand and implement in efficient codes.

References

[1] P. Wolfe, Convergence conditions for ascent methods, SIAM Review 11 (1968) 226–235.
[2] P. Wolfe, Convergence conditions for ascent methods, (II): some corrections, SIAM Review 13 (1971) 185–188.
[3] R. Fletcher, C.M. Reeves, Function minimization by conjugate gradients, Computer Journal 7 (1964) 149–154.
[4] E. Polak, G. Ribière, Note sur la convergence de directions conjuguée, Rev. Francaise Informat Recherche Operationelle, 3e Année 16 (1969) 35–43.
[5] B.T. Polyak, The conjugate gradient method in extreme problems, USSR Computational Mathematics and Mathematical Physics 9 (1969) 94–112.
[6] Y. Liu, C. Storey, Efficient generalized conjugate gradient algorithms, part 1: theory, JOTA–Journal of Optimization Theory and Applications 69 (1991) 129–137.
[7] Y.H. Dai, Y. Yuan, A nonlinear conjugate gradient method with a strong global convergence property, SIAM Journal on Optimization 10 (1999) 177–182.
[8] Y.H. Dai, L.Z. Liao, New conjugate conditions and related nonlinear conjugate gradient methods, Applied Mathematics and Optimization 43 (2001) 87–101.
[9] R. Fletcher, Practical methods of optimization, in: Unconstrained Optimization, Vol. 1, John Wiley & Sons, New York, 1987.
[10] M.R. Hestenes, E.L. Stiefel, Methods of conjugate gradients for solving linear systems, Journal of Research of the National Bureau of Standards 49 (1952) 409–436.
[11] E.M.L. Beale, A derivative of conjugate gradients, in: F.A. Lootsma (Ed.), Numerical Methods for Nonlinear Optimization, Academic Press, London, 1972, pp. 39–43.
[12] M.F. McGuire, P. Wolfe, Evaluating a restart procedure for conjugate gradients, Report RC-4382, IBM Research Center, Yorktown Heights, 1973.
[13] M.J.D. Powell, Nonconvex minimization calculations and the conjugate gradient method, in: Numerical Analysis (Dundee, 1983), in: Lecture Notes in Mathematics, vol. 1066, Springer, Berlin, 1984, pp. 122–141.
[14] N.Y. Deng, Z. Li, Global convergence of three terms conjugate gradient methods, Optimization Method and Software 4 (1995) 273–282.
[15] L. Nazareth, A conjugate direction algorithm without line search, Journal of Optimization Theory and Applications 23 (1977) 373–387.
[16] L. Zhang, W. Zhou, D.H. Li, A descent modified Polak–Ribière–Polyak conjugate gradient method and its global convergence, IMA Journal of Numerical Analysis 26 (2006) 629–640.
[17] L. Zhang, W. Zhou, D.H. Li, Some descent three-term conjugate gradient methods and their global convergence, Optimization Methods and Software 22 (2007) 697–711.
[18] L. Zhang, Y. Zhou, A note on the convergence properties of the original three-term Hestenes–Stiefel method, AMO—Advanced Modeling and Optimization 14 (2012) 159–163.
[19] J. Zhang, Y. Xiao, Z. Wei, Nonlinear conjugate gradient methods with sufficient descent condition for large-scale unconstrained optimization, Mathematical Problems in Engineering 2009 (2009) http://dx.doi.org/10.1155/2009/243290. Article ID 243290.
[20] A.Y. Al-Bayati, W.H. Sharif, A new three-term conjugate gradient method for unconstrained optimization, Canadian Journal on Science and Engineering Mathematics 1 (5) (2010) 108–124.
[21] W. Cheng, A two-term PRP-based descent method, Numerical Functional Analysis and Optimization 28 (2007) 1217–1230.
[22] Y. Narushima, H. Yabe, J.A. Ford, A three-term conjugate gradient method with sufficient descent property for unconstrained optimization, SIAM Journal on Optimization 21 (1) (2011) 212–230.
[23] W.W. Hager, H. Zhang, A new conjugate gradient method with guaranteed descent and an efficient line search, SIAM Journal on Optimization 16 (2005) 170–192.
[24] N. Andrei, A modified Polak–Ribière–Polyak conjugate gradient algorithm for unconstrained optimization, Optimization 60 (2011) 1457–1471.
[25] B.M. Averick, R.G. Carter, J.J. Moré, G.L. Xue, The MINPACK-2 test problem collection, Mathematics and Computer Science Division, Argonne National Laboratory, Preprint MCS-P153-0692, June 1992.
[26] J. Nocedal, Conjugate gradient methods and nonlinear optimization, in: L. Adams, J.L. Nazareth (Eds.), Linear and Nonlinear Conjugate Gradient Related Methods, SIAM, 1996, pp. 9–23.
[27] N. Andrei, Acceleration of conjugate gradient algorithms for unconstrained optimization, Applied Mathematics and Computation 213 (2009) 361–369.
[28] N. Andrei, An acceleration of gradient descent algorithm with backtracking for unconstrained optimization, Numerical Algorithms 42 (2006) 63–73.
[29] N. Andrei, Scaled conjugate gradient algorithms for unconstrained optimization, Computational Optimization and Applications 38 (2007) 401–416.
[30] D.F. Shanno, K.H. Phua, Algorithm 500, minimization of unconstrained multivariate functions, ACM Transactions on Mathematical Software 2 (1976) 87–94.
[31] N. Andrei, Scaled memoryless BFGS preconditioned conjugate gradient algorithm for unconstrained optimization, Optimization Methods and Software 22 (2007) 561–571.
[32] G. Zoutendijk, Nonlinear programming, computational methods, in: J. Abadie (Ed.), Integer and Nonlinear Programming, North-Holland, Amsterdam, 1970, pp. 38–86.
[33] N. Andrei, An unconstrained optimization test functions collection, Advanced Modeling and Optimization 10 (2008) 147–161.
[34] R. Glowinski, Numerical Methods for Nonlinear Variational Problems, Springer-Verlag, Berlin, 1984.
[35] G. Cimatti, On a problem of the theory of lubrication governed by a variational inequality, Applied Mathematics and Optimization 3 (1977) 227–242.
[36] J. Goodman, R. Kohn, L. Reyna, Numerical study of a relaxed variational problem from optimal design, Computer Methods in Applied Mechanics and Engineering 57 (1986) 107–127.
[37] R. Aris, The Mathematical Theory of Diffusion and Reaction in Permeable Catalysts, Oxford, 1975.
[38] J. Bebernes, D. Eberly, Mahematical Problems from Combustion Theory, in: Applied Mathematical Sciences, vol. 83, Springer-Verlag, 1989.
[39] J.C.C. Nitsche, Lectures On Minimal Surfaces, Vol. 1, Cambridge University Press, 1989.