Accelerated conjugate gradient algorithm with modified secant condition for unconstrained optimization

Neculai Andrei
Research Institute for Informatics,
Center for Advanced Modeling and Optimization,
8-10, Averescu Avenue, Bucharest 1, Romania.
Academy of Romanian Scientists,
54, Splaiul Independentei, Bucharest 5, Romania.
E-mail: nandrei@ici.ro

Abstract. Conjugate gradient algorithms are very powerful methods for solving large-scale unconstrained optimization problems characterized by low memory requirements and strong local and global convergence properties. Over 25 variants of different conjugate gradient methods are known. In this paper we propose a fundamentally different method, in which the well known parameter β_k is computed by an approximation of the Hessian / vector product through modified secant condition. For search direction computation, the method takes both the available gradient and the function values information in two successive iteration points and achieves high-order accuracy in approximating the second-order curvature of the minimizing function. For steplength computation the method uses the advantage that the step lengths in conjugate gradient algorithms may differ from 1 by two order of magnitude and tend to vary in an unpredictable manner. Thus, we suggest an acceleration scheme able to improve the efficiency of the algorithm. Under common assumptions, the method is proved to be globally convergent. It is shown that for uniformly convex functions the convergence of the accelerated algorithm is still linear, but the reduction in function values is significantly improved. Numerical comparisons with some conjugate gradient algorithms (including CG_DESCENT by Hager and Zhang [19], CONMIN by Shanno and Phua [29], SCALCG by Andrei [3-5], or LBFGS by Liu and Nocedal [22]) using a set of 750 unconstrained optimization problems, some of them from the CUTE library, show that the suggested algorithm outperforms the known conjugate gradient algorithms and LBFGS.

MSC: 49M07, 49M10, 90C06, 65K
Keywords: Unconstrained optimization, conjugate gradient method, Newton direction, modified secant condition, numerical comparisons

1. Introduction

Let us consider 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. As we know, for solving this problem starting from an initial guess $x_0 \in R^n$ a 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

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

In (1.3) β_k is known as the conjugate gradient parameter, $s_k = x_{k+1} - x_k$ and $g_k = \nabla f(x_k)$. Consider $\|.\|$ the Euclidean norm and define $y_k = g_{k+1} - g_k$. The line search in the conjugate gradient algorithms is often based on the standard Wolfe conditions:

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

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

where d_k is a descent direction and $0<\rho\le\sigma<1$.

The search direction d_k, assumed to be a descent one, plays the main role in these methods. Different conjugate gradient algorithms correspond to different choices for the scalar parameter β_k. On the other hand the stepsize α_k guarantees the global convergence in some cases and is crucial in efficiency. The line search in the conjugate gradient algorithms is often based on the standard Wolfe conditions. Plenty of conjugate gradient methods are known and an excellent survey of these methods with a special attention on their global convergence is given by Hager and Zhang [20]. A numerical comparison of conjugate gradient algorithms (1.2) and (1.3) with Wolfe line search (1.4) and (1.5), for different formulae of parameter β_k computation, including the Dolan and Moré performance profile, is given in [6].

In [23] Jorge Nocedal articulated a number of open problems in conjugate gradient algorithms. Two of them seem to be really very important. One refers to the direction computation in order to take into account the problem structure. In particular, when the problem is partially separable the idea is to use the partitioned updating like in quasi-Newton methods [18]. The second one focuses on the step length. Intensive numerical experiments with conjugate gradient algorithms proved that the step length may differ from 1 up to two orders of magnitude, being larger or smaller than 1, depending on how the problem is scaled. Moreover, the sizes of the step length tend to vary in a totally unpredictable way. This is in contrast with the Newton and quasi-Newton methods, as well as with the limited memory quasi-Newton methods, which usually admit the unit step length for most of the iterations and require only very few function evaluations for step length determination.

In this paper we present a conjugate gradient algorithm which address to these open problems. The structure of the paper is as follows. In section 2 we present a conjugate gradient algorithm with modified secant condition. The idea of this algorithm is to use the Newton direction for β_k computation in (1.3). This leads us to a formula for β_k which contains the Hessian of the minimizing function. In section 3 we present the convergence of the algorithm both for uniformly convex functions and for general nonlinear functions. We prove that under common assumptions and if the direction is a descent one then the method is globally convergent. In section 4 we present an acceleration scheme of the algorithm. The idea of this computational scheme is to take advantage that the step lengths α_k in conjugate gradient algorithms are very different from 1. Therefore, we suggest we modify α_k in such a manner as to improve the reduction of the function values along the iterations. In section 5 we present the ACGMSEC algorithm and we prove that for uniformly convex functions the convergence of the accelerated algorithm is still linear, but the reduction in function values is significantly improved. Numerical comparisons of our algorithm with some other conjugate gradient algorithms including CG_DESCENT by Hager and Zhang [19], CONMIN by Shanno and Phua [29], SCALCG by Andrei [3-5], or limited quasi-Newton LBFGS by Liu and Nocedal [22] are presented in section 6. For this we use a set of 750 unconstrained optimization problems presented in [1], some of them from the CUTE library [10]. We show that the suggested algorithm outperforms the above conjugate gradient algorithms and LBFGS.

2. Conjugate gradient algorithms with modified secant condition

Our motivation to get a good algorithm for solving (1.1) is to choose the parameter β_k in (1.3) in such a way so that for every $k\ge 1$ the direction d_{k+1} given by (1.3) be the Newton direction. Therefore, from the equation

$$-\nabla^2 f(x_{k+1})^{-1} g_{k+1} = -g_{k+1} + \beta_k s_k .$$

after some algebra we get:

$$\beta_k = \frac{s_k^T \nabla^2 f(x_{k+1}) g_{k+1} - s_k^T g_{k+1}}{s_k^T \nabla^2 f(x_{k+1}) s_k}. \tag{2.1}$$

The salient point with this formula for β_k computation is the presence of the Hessian. Observe that if the line search is exact we get the Daniel method [14]. For large-scale problems, choices for the update parameter that do not require the evaluation of the Hessian matrix are often preferred in practice to the methods that require the Hessian. However, the presence of the Hessian in β_k recalls the open problem articulated by Nocedal [23]: whether one can take advantage of the problem structure to design a more efficient nonlinear conjugate gradient iteration. Indeed, our numerical experiments proved that even though the Hessian is partially separable (block diagonal) or it is a multidiagonal matrix, the Hessian / vector product $\nabla^2 f(x_{k+1}) s_k$ is time consuming, especially for large-scale problems. In another effort to use the Hessian in β_k in [8] we experienced a nonlinear conjugate gradient algorithm in which the Hessian / vector product $\nabla^2 f(x_{k+1}) s_k$ is approximated by finite differences. Even though we have got good numerical results, in this paper we prefer to consider another way of using the curvature of the function given by the Hessian.

As we know, for quasi-Newton methods an approximation matrix B_k to the Hessian $\nabla^2 f(x_k)$ is used and updated so that the new matrix B_{k+1} satisfies the secant condition $B_{k+1} s_k = y_k$. Therefore, in order to have an algorithm for solving large-scale problems we can assume that the pair (s_k, y_k) satisfies the secant condition. In this case, Zhang, Deng and Chen [30] proved that if $\|s_k\|$ is sufficiently small, then $s_k^T \nabla^2 f(x_{k+1}) s_k - s_k^T y_k = O(\|s_k\|^3)$. Further, Zhang, Deng and Chen [30] and Zhang and Xu [31] expanded the secant condition and obtained a class of modified secant condition with a vector parameter which uses both the gradients and the function values in two successive points as:

$$B_{k+1} s_k = \hat{y}_k, \quad \hat{y}_k = y_k + \frac{\eta_k}{s_k^T u_k} u_k , \tag{2.2}$$

where

$$\eta_k = 6(f_k - f_{k+1}) + 3(g_k + g_{k+1})^T s_k \tag{2.3}$$

and $u_k \in R^n$ is any vector such that $s_k^T u_k \neq 0$. Obviously, from (2.2) we get

$$s_k^T B_{k+1} s_k = s_k^T y_k + \eta_k . \tag{2.4}$$

Zhang, Deng and Chen [30] proved that if $\|s_k\|$ is sufficiently small, then for any vector u_k with $s_k^T u_k \neq 0$, $s_k^T \nabla^2 f(x_{k+1}) s_k - s_k^T \hat{y}_k = O(\|s_k\|^4)$ holds. Therefore, the quantity $s_k^T \hat{y}_k$ given by the modified secant condition (2.2) approximates the second-order curvature $s_k^T \nabla^2 f(x_{k+1}) s_k$ with a higher precision than the quantity $s_k^T y_k$ does. This is a very good motivation to use it in (2.1). For this purpose, in order to unify both approaches, we consider a slight modification of the modified secant condition (2.2) as $B_{k+1} s_k = z_k$, where

$$z_k = y_k + \frac{\delta \eta_k}{s_k^T u_k} u_k \tag{2.5}$$

and $\delta \geq 0$ is a scalar parameter. With $u_k = s_k$ this leads us to

$$\beta_k = \left(\frac{\delta \eta_k}{\|s_k\|^2} - 1 \right) \frac{s_k^T g_{k+1}}{y_k^T s_k + \delta \eta_k} + \frac{y_k^T g_{k+1}}{y_k^T s_k + \delta \eta_k}. \tag{2.6}$$

Therefore, the direction is

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

which is an approximation of the Newton direction. Observe that (2.7) can be expressed as:

$$d_{k+1} = -Q_{k+1} g_{k+1}, \tag{2.8}$$

where

$$Q_{k+1} = I - \frac{s_k y_k^T}{y_k^T s_k + \delta \eta_k} + \left(1 - \frac{\delta \eta_k}{s_k^T s_k} \right) \frac{s_k s_k^T}{y_k^T s_k + \delta \eta_k} \tag{2.9}$$

is another rank two approximation to the inverse of the Hessian. Since the matrix Q_{k+1} defined by (2.9) is not symmetric and hence not positive definite, therefore the corresponding directions are not necessarily descent and numerical instability can result. For $\delta = 0$ we get exactly the Perry method [24].

3. Convergence analysis

In this section we analyse the convergence of the algorithm (1.2) and (2.7), where $d_0 = -g_0$. In the following we consider that $g_k \neq 0$ for all $k \geq 1$, otherwise a stationary point is obtained. Assume that:

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

(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 $x \in S$. In order to prove the global convergence, we assume that the step size α_k in (1.2) is obtained by the strong Wolfe line search, that is,

$$f(x_k + \alpha_k d_k) - f(x_k) \leq \rho \alpha_k g_k^T d_k, \tag{3.1}$$

$$\left| g_{k+1}^T d_k \right| \leq \sigma g_k^T d_k. \tag{3.2}$$

where ρ and σ are positive constants such that $0 < \rho \leq \sigma < 1$.

Dai *et al.* [13] proved that for any conjugate gradient method with strong Wolfe line search the following general result holds:

Lemma 3.1. *Suppose that the assumptions (i) and (ii) hold and consider any conjugate gradient method (1.2) and (1.3), where* d_k *is a descent direction and* α_k *is obtained by the strong Wolfe line search (3.1) and (3.2). If*

$$\sum_{k \geq 1} \frac{1}{\|d_k\|^2} = \infty, \tag{3.3}$$

then

$$\liminf_{k \to \infty} \|g_k\| = 0. \ \blacksquare \tag{3.4}$$

To prove the global convergence of the algorithm we need the following estimates. Using the mean value theorem in (2.3) we have:

$$\eta_k = 6(f_k - f_{k+1}) + 3(g_k + g_{k+1})^T s_k$$

$$= 6\nabla f(\xi_k)^T(x_k - x_{k+1}) + 3(\nabla f(x_k) + \nabla f(x_{k+1}))^T s_k$$
$$= -3\nabla f(\xi_k)^T s_k - 3\nabla f(\xi_k)^T s_k + 3\nabla f(x_k)^T s_k + 3\nabla f(x_{k+1})^T s_k$$
$$= 3\left(\nabla f(x_k) - \nabla f(\xi_k) + \nabla f(x_{k+1}) - \nabla f(\xi_k)\right)^T s_k,$$

where $\xi_k = \tau x_k + (1-\tau)x_{k+1}$ and $\tau \in (0,1)$. From the Lipschitz continuity we have:

$$|\eta_k| \le 3\left(\|\nabla f(x_k) - \nabla f(\xi_k)\| + \|\nabla f(x_{k+1}) - \nabla f(\xi_k)\|\right)\|s_k\|$$
$$\le 3\left(L\|x_k - \xi_k\| + L\|x_{k+1} - \xi_k\|\right)\|s_k\|$$
$$= 3\left(L(1-\tau)\|x_k - x_{k+1}\| + L\tau\|x_{k+1} - x_k\|\right)\|s_k\|$$
$$= 3L(1-\tau)\|s_k\|^2 + 3L\tau\|s_k\|^2 = 3L\|s_k\|^2. \tag{3.5}$$

On the other hand

$$|y_k^T s_k + \delta\eta_k| \le |y_k^T s_k| + \delta|\eta_k|$$
$$\le \|y_k\|\|s_k\| + \delta|\eta_k| \le L\|s_k\|^2 + 3\delta L\|s_k\|^2 = L(1+3\delta)\|s_k\|^2. \tag{3.6}$$

Global convergence for uniformly convex functions. For uniformly convex functions which satisfy the above assumptions *(i)* and *(ii)* we can prove that the norm of d_{k+1} generated by (2.7) is bounded above. Thus, by Lemma 3.1 we can prove the global convergence of the algorithm (1.2) and (2.7).

As we know, if f is a uniformly convex function, then there exists a constant $\mu > 0$ such that

$$(\nabla f(x) - \nabla f(y))^T(x - y) \ge \mu\|x - y\|^2, \text{ for any } x, y \in S. \tag{3.7}$$

Equivalently, this can be expressed as

$$f(x) \ge f(y) + \nabla f(y)^T(x - y) + \frac{\mu}{2}\|x - y\|^2, \text{ for any } x, y \in S. \tag{3.8}$$

From (3.7) and (3.8) it follows that

$$y_k^T s_k \ge \mu\|s_k\|^2, \tag{3.9}$$

$$f_k - f_{k+1} \ge -g_{k+1}^T s_k + \frac{\mu}{2}\|s_k\|^2. \tag{3.10}$$

Obviously, from (3.9) and (3.10) we get:

$$\mu\|s_k\|^2 \le y_k^T s_k \le L\|s_k\|^2, \tag{3.11}$$

i.e. $\mu \le L$.

Theorem 3.1. *Suppose that the assumptions (i) and (ii) hold and f is a uniformly convex function. Consider the algorithm (1.2) and (2.7), where d_{k+1} is a descent direction and α_k is obtained by the strong Wolfe line search (3.1) and (3.2). If $L = \mu$, then for any $\delta \ge 0$ the algorithm satisfies $\lim_{k\to\infty} g_k = 0$. If $L > \mu$, then for $0 \le \delta \le L/(3(L-\mu))$ the algorithm satisfies $\lim_{k\to\infty} g_k = 0$.*

Proof. Using the above relations (3.10) and (3.11) we have

$$y_k^T s_k + \delta\eta_k = y_k^T s_k + 6\delta(f_k - f_{k+1}) + 3\delta(g_k + g_{k+1})^T s_k$$
$$\ge y_k^T s_k + 6\delta(-g_{k+1}^T s_k + \frac{\mu}{2}\|s_k\|^2) + 3\delta(g_k + g_{k+1})^T s_k$$
$$= y_k^T s_k - 6\delta g_{k+1}^T s_k + 3\delta\mu\|s_k\|^2 + 3\delta g_k^T s_k + 3\delta g_{k+1}^T s_k$$

$$= (1-3\delta) y_k^T s_k + 3\delta\mu \|s_k\|^2 \geq (1-3\delta) y_k^T s_k + \frac{3\delta\mu}{L} y_k^T s_k$$

$$= (1-3\delta+\frac{3\delta\mu}{L}) y_k^T s_k. \tag{3.12}$$

Now, if $L = \mu,$ then for all $\delta \geq 0$, $y_k^T s_k + \delta\eta_k \geq \mu\|s_k\|^2$, i.e. $y_k^T s_k + \delta\eta_k \geq m\|s_k\|^2$, where $m = \mu.$

On the other hand, if $L \geq \mu,$ then for $0 \leq \delta < \frac{L}{3(L-\mu)}$, the coefficient of the right hand side of (3.12) is positive, that is $y_k^T s_k + \delta\eta_k \geq (1-3\delta+\frac{3\delta\mu}{L})\mu\|s_k\|^2$, i.e. $y_k^T s_k + \delta\eta_k \geq m\|s_k\|^2$, where $m = (1-3\delta+\frac{3\delta\mu}{L})\mu.$

Therefore,

$$\|d_{k+1}\| = \left\| -g_{k+1} + \frac{y_k^T g_{k+1}}{y_k^T s_k + \delta\eta_k} s_k - \left(1 - \frac{\delta\eta_k}{\|s_k\|^2}\right) \frac{s_k^T g_{k+1}}{y_k^T s_k + \delta\eta_k} s_k \right\|$$

$$\leq \|g_{k+1}\| + \frac{\|y_k\|\|g_{k+1}\|}{|y_k^T s_k + \delta\eta_k|}\|s_k\| + \left|1 - \frac{\delta\eta_k}{\|s_k\|^2}\right| \frac{\|s_k\|\|g_{k+1}\|}{|y_k^T s_k + \delta\eta_k|}\|s_k\|. \tag{3.13}$$

But, from (3.5) it follows that

$$\left|1 - \frac{\delta\eta_k}{\|s_k\|^2}\right| \leq 1 + \frac{\delta|\eta_k|}{\|s_k\|^2} \leq 1 + \frac{\delta 3L\|s_k\|^2}{\|s_k\|^2} = 1 + 3\delta L. \tag{3.14}$$

From (3.13), having in view the Lipschitz continuity, (3.14) and the above estimation on $y_k^T s_k + \delta\eta_k$ we get:

$$\|d_{k+1}\| \leq \|g_{k+1}\| + \frac{L\|g_{k+1}\|}{m\|s_k\|^2}\|s_k\|^2 + \left|1 - \frac{\delta\eta_k}{\|s_k\|^2}\right| \frac{\|g_{k+1}\|}{m\|s_k\|^2}\|s_k\|^2$$

$$\leq \|g_{k+1}\| + \frac{L}{m}\|g_{k+1}\| + \frac{1+3\delta L}{m}\|g_{k+1}\|$$

$$\leq (1 + \frac{L}{m} + \frac{1+3\delta L}{m})\Gamma. \tag{3.15}$$

This relation shows that

$$\sum_{k\geq 1} \frac{1}{\|d_k\|^2} \geq \left(\frac{m}{(m+L+1+3\delta L)\Gamma}\right)^2 \sum_{k\geq 1} 1 = \infty.$$

Therefore, from Lemma 3.1 we have $\liminf_{k\to\infty} \|g_k\| = 0,$ which for uniformly convex functions is equivalent to $\lim_{k\to\infty} g_k = 0.$ ■

Observe that for $L > \mu$, $\frac{L}{3(L-\mu)} > \frac{1}{3}$. Theorem 3.1 says that there is a constant $\bar{\delta} > 1/3$ such that for any $0 \leq \delta \leq \bar{\delta}$ we have $\lim_{k\to\infty} g_k = 0.$ In case of a given problem both constants L and μ are not evaluated. Therefore, we do not know how to estimate the parameter δ or whether $\delta = 1$ can be taken. Yet, $\delta = 0$ is admissible.

Global convergence for general nonlinear functions. For general nonlinear functions, following the method of Dai and Liao [11] or that of Yabe and Takano [32], we replace (2.6) by:

$$\beta_k^+ = \max\left\{\frac{y_k^T g_{k+1}}{y_k^T s_k + \delta\eta_k}, 0\right\} - \left(1 - \frac{\delta\eta_k}{\|s_k\|^2}\right)\frac{s_k^T g_{k+1}}{y_k^T s_k + \delta\eta_k} \tag{3.16}$$

and prove that the corresponding algorithm with strong Wolfe line search is globally convergent. Assume that the direction d_{k+1} satisfies the descent condition

$$g_{k+1}^T d_{k+1} \le 0. \tag{3.17}$$

To prove the global convergence by contradiction we assume that there is a positive constant $\overline{\gamma}$ such that

$$\|g_k\| \ge \overline{\gamma} \text{ for all } k \ge 0. \tag{3.18}$$

Our convergence analysis of (1.2) and (3.16) for general nonlinear functions follows the insights developed by Gilbert and Nocedal in their analysis of the PRP+ conjugate gradient scheme [16] or those given by Hager and Zhang of their CG_DESCENT algorithm [19]. Similar to the approach of Yabe and Takano [32] we establish a bound for the change $w_{k+1} - w_k$ in the normalized direction $w_k = d_k / \|d_k\|$, which is used to conclude that the gradients cannot be bounded away from zero.

Lemma 3.2. *Suppose that the assumptions (i) and (ii) hold and consider the conjugate gradient algorithm (1.2), where the direction d_{k+1} given by (1.3) and (3.16) satisfies the descent condition (3.17) and α_k is obtained by the strong Wolfe line search conditions (3.1) and (3.2). If (3.18) holds and δ is chosen so that*

$$0 \le \delta < \frac{1-\sigma}{3(1+\sigma-2\rho)}$$

then $d_{k+1} \ne 0$ and

$$\sum_{k\ge 1} \|w_{k+1} - w_k\|^2 < \infty, \tag{3.19}$$

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

Proof. The proof is similar to that of Lemma 4 given in Yabe and Takano [32]. Obviously, by (3.17) we have $d_k \ne 0$. Therefore, w_k is well defined. Now, from (3.18) and from Lemma 3.1 it follows that

$$\sum_{k\ge 0} \frac{1}{\|d_k\|^2} < \infty,$$

otherwise (3.4) holds, contradicting (3.18). In the following we write:

$$\beta_k^+ = \beta_k^1 + \beta_k^2, \tag{3.20}$$

where:

$$\beta_k^1 = \max\left\{\frac{y_k^T g_{k+1}}{y_k^T s_k + \delta\eta_k}, 0\right\}, \tag{3.21}$$

$$\beta_k^2 = -\left(1 - \frac{\delta\eta_k}{\|s_k\|^2}\right)\frac{s_k^T g_{k+1}}{y_k^T s_k + \delta\eta_k}. \tag{3.22}$$

Define:

$$v_{k+1} = -g_{k+1} + \beta_k^2 s_k, \tag{3.23}$$

$$r_{k+1} = \frac{v_{k+1}}{\|d_{k+1}\|}, \tag{3.24}$$

$$\tau_{k+1} = \beta_k^1 \frac{\|d_k\|}{\|d_{k+1}\|} \geq 0. \tag{3.25}$$

Therefore, we have

$$\begin{aligned} w_{k+1} &= \frac{d_{k+1}}{\|d_{k+1}\|} = \frac{-g_{k+1} + \beta_k^1 s_k + \beta_k^2 s_k}{\|d_{k+1}\|} \\ &= \frac{-g_{k+1} + \beta_k^2 s_k}{\|d_{k+1}\|} + \beta_k^1 \frac{\|d_k\|}{\|d_{k+1}\|} \frac{s_k}{\|d_k\|} \\ &= r_{k+1} + \tau_{k+1}\alpha_k w_k. \end{aligned}$$

Now, since $\|w_k\| = \|w_{k+1}\| = 1,$ it follows that

$$\begin{aligned} \|r_{k+1}\|^2 &= \|w_{k+1} - \tau_{k+1}\alpha_k w_k\|^2 = \|w_{k+1}\|^2 - 2\tau_{k+1}\alpha_k w_{k+1}^T w_k + \tau_{k+1}^2\alpha_k^2 \|w_k\|^2 \\ &= \|w_k\|^2 - 2\tau_{k+1}\alpha_k w_{k+1}^T w_k + \tau_{k+1}^2\alpha_k^2 \|w_{k+1}\|^2 = \|\tau_{k+1}\alpha_k w_{k+1} - w_k\|^2. \end{aligned}$$

Therefore,

$$\|r_{k+1}\| = \|w_{k+1} - \tau_{k+1}\alpha_k w_k\| = \|\tau_{k+1}\alpha_k w_{k+1} - w_k\|.$$

Since $\tau_{k+1} \geq 0$ we get

$$\begin{aligned} \|w_{k+1} - w_k\| &\leq \|(1 + \tau_{k+1}\alpha_k)(w_{k+1} - w_k)\| \\ &= \|w_{k+1} + \tau_{k+1}\alpha_k w_{k+1} - w_k - \tau_{k+1}\alpha_k w_k\| \\ &\leq \|w_{k+1} - \tau_{k+1}\alpha_k w_k\| + \|\tau_{k+1}\alpha_k w_{k+1} - w_k\| = 2\|r_{k+1}\|. \end{aligned} \tag{3.26}$$

Now, we evaluate the quantity $y_k^T s_k + \delta\eta_k.$ Using the strong Wolfe conditions we have:

$$\begin{aligned} y_k^T s_k + \delta\eta_k &= y_k^T s_k + 6\delta(f_k - f_{k+1}) + 3\delta(g_k + g_{k+1})^T s_k \\ &\geq y_k^T s_k - 6\delta\rho g_k^T s_k + 3\delta(g_k + g_{k+1})^T s_k \\ &= (g_{k+1} - g_k)^T s_k - 6\delta\rho g_k^T s_k + 3\delta(g_k + g_{k+1})^T s_k \\ &= (1 + 3\delta) g_{k+1}^T s_k + (3\delta - 6\delta\rho - 1) g_k^T s_k \\ &\geq (1 + 3\delta)\sigma g_k^T s_k + (3\delta - 6\delta\rho - 1) g_k^T s_k \\ &= \left[3(1 + \sigma - 2\rho)\delta - (1 - \sigma)\right] g_k^T s_k. \end{aligned} \tag{3.27}$$

We assumed that $g_k^T s_k = \alpha_k g_k^T d_k < 0.$ Therefore, if $0 \leq \delta < \frac{1-\sigma}{3(1+\sigma-2\rho)},$ then there is a constant $M > 0$ such that

$$y_k^T s_k + \delta\eta_k \geq -M g_k^T s_k > 0. \tag{3.28}$$

From the definition of v_{k+1} it follows that

$$\begin{aligned} \|v_{k+1}\| &= \|-g_{k+1} + \beta_k^2 s_k\| \leq \|g_{k+1}\| + |\beta_k^2| \|s_k\| \\ &= \|g_{k+1}\| + \left|1 - \frac{\delta\eta_k}{\|s_k\|^2}\right| \frac{|s_k^T g_{k+1}|}{|y_k^T s_k + \delta\eta_k|} \|s_k\| \\ &\leq \|g_{k+1}\| + \left|1 - \frac{\delta\eta_k}{\|s_k\|^2}\right| \frac{\sigma |s_k^T g_k|}{M |s_k^T g_k|} \|s_k\|. \end{aligned}$$

Therefore, using (3.14) we have

$$\|v_{k+1}\| \le \|g_{k+1}\| + (1+3L\delta)\frac{\sigma}{M}\|s_k\| \le \Gamma + (1+3L\delta)\frac{\sigma}{M}D. \tag{3.29}$$

With the above estimates we get:

$$\sum_{k\ge 1}\|w_{k+1}-w_k\|^2 = \sum_{k\ge 1} 4\|r_k\|^2 = 4\sum_{k\ge 1}\frac{\|v_k\|^2}{\|d_k\|^2}$$

$$\le 4\left(\Gamma + (1+3L\delta)\frac{\sigma}{M}D\right)^2 \sum_{k\ge 1}\frac{1}{\|d_k\|^2} < \infty,$$

i.e. (3.19) holds, which completes the proof. ■

This Lemma shows that asymptotically the search directions generated by the algorithm change slowly. Using Lemma 3.2 and assuming that d_k satisfies the sufficient descent condition

$$g_k^T d_k \le -c\|g_k\|^2, \tag{3.30}$$

where $c > 0$ is a constant, we can establish the following Lemma, showing that β_k^+ satisfies a slightly different form of *Property (*)*. The Property (*), first derived by Gilbert and Nocedal [16], shows that β_k in conjugate gradient algorithms will be small when the step s_k is small. For example, β_k^{PRP} has this property, this explaining the efficiency of the PRP conjugate gradient algorithm. Suppose that the step length α_k obtained by the strong Wolfe conditions (3.1) and (3.2) is bounded away from zero, i.e. there is a positive constant $\omega > 0$ such that $\alpha_k \ge \omega$.

Lemma 3.3. *Suppose that the assumptions (i) and (ii) hold and consider the conjugate gradient algorithm (1.2), where the direction d_{k+1} given by (1.3) and (3.16) satisfies the sufficient descent condition (3.30) and α_k is obtained by the strong Wolfe line search conditions (3.1) and (3.2) and $\alpha_k \ge \omega$. If $0 \le \delta < \dfrac{1-\sigma}{3(1+\sigma-2\rho)}$ then there exist the constants $b > 1$ and $\xi > 0$ such that*

$$|\beta_k^+| \le b \tag{3.31}$$

and

$$\|s_k\| \le \xi \Rightarrow |\beta_k^+| \le \frac{1}{b} \tag{3.32}$$

for all k.

Proof. From (3.28), (3.30) and using (3.18) we get:

$$y_k^T s_k + \delta\eta_k \ge -Mg_k^T s_k \ge Mc\omega\|g_k\|^2 \ge Mc\omega\bar{\gamma}^2. \tag{3.33}$$

Now, from (3.16), using (3.14) and (3.33) we have:

$$|\beta_k^+| \le \left|\frac{y_k^T g_{k+1}}{y_k^T s_k + \delta\eta_k}\right| + \left|1 - \frac{\delta\eta_k}{\|s_k\|^2}\right|\left|\frac{s_k^T g_{k+1}}{y_k^T s_k + \delta\eta_k}\right|$$

$$\le \frac{|y_k^T g_{k+1}| + (1+3\delta L)|s_k^T g_{k+1}|}{Mc\omega\bar{\gamma}^2}$$

$$\leq \frac{\|y_k\|\|g_{k+1}\| + (1+3\delta L)\|s_k\|\|g_{k+1}\|}{Mc\omega\bar{\gamma}^2}$$
$$\leq \frac{L+1+3\delta L}{Mc\omega\bar{\gamma}^2}\|s_k\|\|g_{k+1}\| \leq \frac{L+1+3\delta L}{Mc\omega\bar{\gamma}^2} D\Gamma$$
$$= \frac{(L+1+3\delta L)D\Gamma}{Mc\omega\bar{\gamma}^2} \equiv b. \tag{3.34}$$

Without loss of generality we can define b such that $b > 1$. Let us define:

$$\xi \equiv \left(\frac{Mc\omega\bar{\gamma}^2}{(L+1+3\delta L)\Gamma} \right)^2 \frac{1}{D}. \tag{3.35}$$

Obviously, if $\|s_k\| \leq \xi$, from the fourth inequality in (3.34) we have

$$\left|\beta_k^+\right| \leq \frac{(L+1+3\delta L)\Gamma}{Mc\omega\bar{\gamma}^2}\xi = \frac{1}{b}.$$

Therefore, for b and ξ defined in (3.34) and (3.35) respectively, (3.31) and (3.32) hold. ■

The Property (*) presented in Lemma 3.3 can be used to show that if the gradients are bounded away from zero and (3.31) and (3.32) hold, then a finite number of steps s_k cannot be too small. Therefore, the algorithm makes a rapid progress to the optimum. Indeed, for $\lambda > 0$ and a positive integer Δ let us define the set of indices:

$$K_{k,\Delta}^{\lambda} = \left\{ i \in N^* : k \leq i \leq k+\Delta-1, \|s_{i-1}\| > \lambda \right\},$$

where N^* is the set of positive integers. The following Lemma is similar to Lemma 3.5 in Dai and Liao [11] and to Lemma 4.2 in Gilbert and Nocedal [16].

Lemma 3.4. *Suppose that all the assumptions of Lemma 3.3 are satisfied. Then there is a* $\lambda > 0$ *such that for any* $\Delta \in N^*$ *and any index* k_0*, there is an index* $k \geq k_0$ *such that* $\left|K_{k,\Delta}^{\lambda}\right| > \Delta/2$.

Using Lemma 3.2 and Lemma 3.4 we can prove the global convergence theorem for method (1.2), (1.3) and (3.16). The theorem is similar to Theorem 3.6 in Dai and Liao [11] or to Theorem 3.2 in Hager and Zhang [19] and the proof is omitted here.

Theorem 3.2. *Suppose that the assumptions (i) and (ii) hold and consider the conjugate gradient algorithm (1.2), where the direction* d_{k+1} *given by (1.3) and (3.16) satisfies the sufficient descent condition (3.30) and* α_k *is obtained by the strong Wolfe line search conditions (3.1) and (3.2). If* $0 \leq \delta < \dfrac{1-\sigma}{3(1+\sigma-2\rho)}$ *then* $\liminf_{k\to\infty} \|g_k\| = 0$. ■

Since ρ and σ are given in the Wolfe line search conditions, it follows that the upper bound of δ established in Theorem 3.2 is smaller than 1/3. Again observe that $\delta = 0$ is admissible. Even though the modified secant condition (2.2), as given by Zhang, Deng and Chen [] and Zhang and Xu [], has $\delta = 1$, we do not know whether this value for δ can be considered in numerical experiments.

Although we were able to prove the global convergence of the computational scheme (1.2), (1.3) and (3.16), however, its computational performances are greatly improved by an acceleration scheme which we present in the next section.

4. Acceleration of the algorithm

It is common to see 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. In order to improve the performances of the conjugate gradient algorithms the efforts were directed to design procedures for direction computation based on the second order information. For example, CONMIN [], and SCALCG [] take this idea of BFGS preconditioning. In this section we focus on the step length modification. In the context of gradient descent algorithm with backtracking the step length modification has been considered for the first time in [2].

Jorge Nocedal [23] pointed out that in conjugate gradient methods the step lengths may differ from 1 in a very unpredictable manner. They can be larger or smaller than 1 depending on how the problem is scaled. Numerical comparisons between conjugate gradient methods and the limited memory quasi Newton method, by Liu and Nocedal [22], show that the latter is more successful [6]. One explanation of the efficiency of the limited memory quasi-Newton method is given by its ability to accept unity step lengths along the iterations. In this section we take advantage of this behavior of conjugate gradient algorithms and present an acceleration scheme. Basically this modifies the step length in a multiplicative manner to improve the reduction of the function values along the iterations.

Line search. For implementing the algorithm (1.2) one of the crucial elements is the stepsize computation. In the following we consider the line searches that satisfy either the Goldstein's conditions [17]:

$$\rho_1 \alpha_k g_k^T d_k \le f(x_k + \alpha_k d_k) - f(x_k) \le \rho_2 \alpha_k g_k^T d_k, \tag{4.1}$$

where $0 < \rho_2 < \frac{1}{2} < \rho_1 < 1$ and $\alpha_k > 0,$ or the Wolfe conditions (1.4) and (1.5).

Proposition 4.1. *Assume that d_k is a descent direction and ∇f satisfies the Lipschitz condition $\|\nabla f(x) - \nabla f(x_k)\| \le L\|x - x_k\|$ 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 Goldstein conditions (4.1), then*

$$\alpha_k \ge \frac{(1-\rho_1)}{L} \frac{\left|g_k^T d_k\right|}{\|d_k\|^2}. \tag{4.2}$$

If the line search satisfies the Wolfe conditions (1.4) and (1.5), then

$$\alpha_k \ge \frac{(1-\sigma)}{L} \frac{\left|g_k^T d_k\right|}{\|d_k\|^2}. \tag{4.3}$$

Proof. If the Goldstein conditions are satisfied, then using the mean value theorem from (4.1) we get:

$$\begin{aligned} \rho_1 \alpha_k g_k^T d_k &\le f(x_k + \alpha_k d_k) - f(x_k) \\ &= \alpha_k \nabla f(x_k + \xi d_k)^T d_k \le \alpha_k g_k^T d_k + L\alpha_k^2 \|d_k\|^2, \end{aligned}$$

where $\xi \in [0, \alpha_k]$. From this inequality we immediately get (4.2).

Now, to prove (4.3) subtract $g_k^T d_k$ from both sides of (1.5) and using the Lipschitz condition we get:

$$(\sigma - 1) g_k^T d_k \le (g_{k+1} - g_k)^T d_k \le \alpha_k L \|d_k\|^2. \tag{4.4}$$

But, d_k is a descent direction and since $\sigma < 1$, we immediately get (4.3). ■

Therefore, satisfying the Goldstein or the Wolfe line search conditions α is bounded away from zero, i.e. there exists a positive constant ω, such that $\alpha \geq \omega$.

Acceleration scheme [7]. Given the initial point x_0 we can compute $f_0 = f(x_0)$, $g_0 = \nabla f(x_0)$ and by Wolfe line search conditions (1.4) and (1.5) the steplength α_0 is determined. With these, the next iteration is computed as: $x_1 = x_0 + \alpha_0 d_0$, ($d_0 = -g_0$) where f_1 and g_1 are immediately determined and the direction d_1 can be computed as: $d_1 = -g_1 + \beta_0 d_0$, where the conjugate gradient parameter β_0 is computed as in (3.16) with a given value for δ. Therefore, at the iteration $k = 1, 2, \ldots$ we know x_k, f_k, g_k and $d_k = -g_k + \beta_{k-1} d_{k-1}$. Suppose that d_k is a descent direction. By the Wolfe line search (1.4) and (1.5) we can compute α_k with which the following point $z = x_k + \alpha_k d_k$ is determined. The first Wolfe condition (1.4) shows that the steplength $\alpha_k > 0$ satisfies:

$$f(z) = f(x_k + \alpha_k d_k) \leq f(x_k) + \rho \alpha_k g_k^T d_k.$$

With these, let us introduce the accelerated conjugate gradient algorithm by means of the following iterative scheme:

$$x_{k+1} = x_k + \gamma_k \alpha_k d_k, \tag{4.5}$$

where $\gamma_k > 0$ is a parameter which follows to be determined in such a manner as to improve the behavior of the algorithm. Now, we have:

$$f(x_k + \alpha_k d_k) = f(x_k) + \alpha_k g_k^T d_k + \frac{1}{2}\alpha_k^2 d_k^T \nabla^2 f(x_k) d_k + o\left(\left\|\alpha_k d_k\right\|^2\right). \tag{4.6}$$

On the other hand, for $\gamma > 0$ we have:

$$f(x_k + \gamma\alpha_k d_k) = f(x_k) + \gamma\alpha_k g_k^T d_k + \frac{1}{2}\gamma^2\alpha_k^2 d_k^T \nabla^2 f(x_k) d_k + o\left(\left\|\gamma\alpha_k d_k\right\|^2\right). \tag{4.7}$$

With these we can write:

$$f(x_k + \gamma\alpha_k d_k) = f(x_k + \alpha_k d_k) + \Psi_k(\gamma), \tag{4.8}$$

where

$$\begin{aligned}\Psi_k(\gamma) = &\frac{1}{2}(\gamma^2 - 1)\alpha_k^2 d_k^T \nabla^2 f(x_k) d_k + (\gamma - 1)\alpha_k g_k^T d_k \\ &+ \gamma^2 \alpha_k o\left(\alpha_k \left\|d_k\right\|^2\right) - \alpha_k o\left(\alpha_k \left\|d_k\right\|^2\right).\end{aligned} \tag{4.9}$$

Let us denote:

$$\begin{aligned} a_k &= \alpha_k g_k^T d_k \leq 0, \\ b_k &= \alpha_k^2 d_k^T \nabla^2 f(x_k) d_k, \\ \varepsilon_k &= o\left(\alpha_k \left\|d_k\right\|^2\right). \end{aligned}$$

Observe that $a_k \leq 0$, since d_k is a descent direction, and for convex functions $b_k \geq 0$. Therefore,

$$\Psi_k(\gamma) = \frac{1}{2}(\gamma^2 - 1)b_k + (\gamma - 1)a_k + \gamma^2\alpha_k\varepsilon_k - \alpha_k\varepsilon_k. \tag{4.10}$$

Now, we see that $\Psi'_k(\gamma) = (b_k + 2\alpha_k\varepsilon_k)\gamma + a_k$ and $\Psi'_k(\gamma_m) = 0$ where

$$\gamma_m = -\frac{a_k}{b_k + 2\alpha_k\varepsilon_k}. \tag{4.11}$$

Observe that $\Psi_k'(0) = a_k < 0$. Therefore, assuming that $b_k + 2\alpha_k\varepsilon_k > 0$, then $\Psi_k(\gamma)$ is a convex quadratic function with minimum value in point γ_m and

$$\Psi_k(\gamma_m) = -\frac{(a_k + (b_k + 2\alpha_k\varepsilon_k))^2}{2(b_k + 2\alpha_k\varepsilon_k)} \leq 0.$$

Considering $\gamma = \gamma_m$ in (4.8) and since $b_k \geq 0$, we see that for every k

$$f(x_k + \gamma_m\alpha_k d_k) = f(x_k + \alpha_k d_k) - \frac{(a_k + (b_k + 2\alpha_k\varepsilon_k))^2}{2(b_k + 2\alpha_k\varepsilon_k)} \leq f(x_k + \alpha_k d_k),$$

which is a possible improvement of the values of function f (when $a_k + (b_k + 2\alpha_k\varepsilon_k) \neq 0$). Therefore, using this simple multiplicative modification of the stepsize α_k as $\gamma_k\alpha_k$ where $\gamma_k = \gamma_m = -a_k/(b_k + 2\alpha_k\varepsilon_k)$ we get:

$$f(x_{k+1}) = f(x_k + \gamma_k\alpha_k d_k) \leq f(x_k) + \rho\alpha_k g_k^T d_k - \frac{(a_k + (b_k + 2\alpha_k\varepsilon_k))^2}{2(b_k + 2\alpha_k\varepsilon_k)}$$

$$= f(x_k) - \left[\frac{(a_k + (b_k + 2\alpha_k\varepsilon_k))^2}{2(b_k + 2\alpha_k\varepsilon_k)} - \rho a_k\right] \leq f(x_k), \qquad (4.12)$$

since $a_k \leq 0$, (d_k is a descent direction).

Observe that if d_k is a descent direction, then

$$\frac{(a_k + (b_k + 2\alpha_k\varepsilon_k))^2}{2(b_k + 2\alpha_k\varepsilon_k)} > \frac{(a_k + b_k)^2}{2b_k}$$

and from (4.12) we get:

$$f(x_{k+1}) \leq f(x_k) - \left[\frac{(a_k + (b_k + 2\alpha_k\varepsilon_k))^2}{2(b_k + 2\alpha_k\varepsilon_k)} - \rho a_k\right]$$

$$< f(x_k) - \left[\frac{(a_k + b_k)^2}{2b_k} - \rho a_k\right] \leq f(x_k).$$

Therefore, neglecting the contribution of ε_k, we still get an improvement on the function values.

Now, in order to get the algorithm we have to determine a way for b_k computation. For this, at point $z = x_k + \alpha_k d_k$ we have:

$$f(z) = f(x_k + \alpha_k d_k) = f(x_k) + \alpha_k g_k^T d_k + \frac{1}{2}\alpha_k^2 d_k^T \nabla^2 f(\tilde{x}_k) d_k,$$

where $\tilde{x}_k$ is a point on the line segment connecting x_k and z. On the other hand, at point $x_k = z - \alpha_k d_k$ we have:

$$f(x_k) = f(z - \alpha_k d_k) = f(z) - \alpha_k g_z^T d_k + \frac{1}{2}\alpha_k^2 d_k^T \nabla^2 f(\bar{x}_k) d_k,$$

where $g_z = \nabla f(z)$ and $\bar{x}_k$ is a point on the line segment connecting x_k and z. Having in view the local character of searching and that the distance between x_k and z is small enough, we can consider $\tilde{x}_k = \bar{x}_k = x_k$. So, adding the above equalities we get:

$$b_k = -\alpha_k y_k^T d_k, \qquad (4.13)$$

where $y_k = g_k - g_z$.

Observe that if $|a_k| > b_k$, then $\gamma_k > 1$. In this case $\gamma_k \alpha_k > \alpha_k$ and it is also possible that $\gamma_k \alpha_k \le 1$ or $\gamma_k \alpha_k > 1$. Hence, the steplength $\gamma_k \alpha_k$ can be greater than 1. On the other hand, if $|a_k| \le b_k$, then $\gamma_k \le 1$. In this case $\gamma_k \alpha_k \le \alpha_k$, so the steplength $\gamma_k \alpha_k$ is reduced. Therefore, if $|a_k| \ne b_k$, then $\gamma_k \ne 1$ and the steplength α_k computed by Wolfe conditions will be modified by its increasing or its reducing through factor γ_k.

Neglecting ε_k in (4.10), we see that $\Psi_k(1) = 0$ and if $|a_k| \le b_k / 2$, then $\Psi_k(0) = -a_k - b_k / 2 \le 0$ and $\gamma_k < 1$. Therefore, for any $\gamma \in [0,1]$, $\Psi_k(\gamma) \le 0$. As a consequence for any $\gamma \in (0,1)$, it follows that $f(x_k + \gamma \alpha_k d_k) < f(x_k)$. In this case, for any $\gamma \in [0,1]$, $\gamma_k \alpha_k \le \alpha_k$. However, in our algorithm we selected $\gamma_k = \gamma_m$ as the point achieving the minimum value of $\Psi_k(\gamma)$.

5. Algorithm ACGMSEC

Using the above developments the following accelerated conjugate gradient algorithm with modified secant condition can be presented.

Step 1. Select the initial starting point $x_0 \in dom\, f$ and compute: $f_0 = f(x_0)$ and $g_0 = \nabla f(x_0)$. Set $d_0 = -g_0$ and $k = 0$. Select a value for parameters ε and τ.

Step 2. Test a criterion for stopping the iterations. For example, if $\|g_k\|_\infty \le \varepsilon$, then stop; otherwise continue with step 3.

Step 3. Using the Wolfe line search conditions (1.4) and (1.5) determine the steplength α_k.

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: $a_k = \alpha_k g_k^T d_k$, and $b_k = -\alpha_k y_k^T d_k$.

Step 6. If $b_k \ne 0$, then compute $\gamma_k = -a_k / b_k$ and update the variables as $x_{k+1} = x_k + \gamma_k \alpha_k d_k$, 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. Set $\delta = 0$. If $\|s_k\| \le \tau$, then set $\delta = 1$.

Step 8. Determine β_k as in (3.16).

Step 9. Compute the search direction as: $d_{k+1} = -g_{k+1} + \beta_k s_k$.

Step 10. Restart criterion. If the restart criterion of Powell $|g_{k+1}^T g_k| > 0.2 \|g_{k+1}\|^2$ is satisfied, then set $d_{k+1} = -g_{k+1}$.

Step 11. Compute the initial guess $\alpha_k = \alpha_{k-1} \|d_{k-1}\| / \|d_k\|$, set $k = k + 1$ and continue with step 2. ■

It is well known that 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, then we restart the algorithm with the negative gradient $-g_{k+1}$. Under reasonable assumptions, the Wolfe conditions and the Powell restart criterion are sufficient to prove the global convergence of the algorithm. The first trial of the step length crucially affects the practical behavior of the algorithm. At every iteration $k \ge 1$

the starting guess for the step α_k in the line search is computed as $\alpha_{k-1}\|d_{k-1}\|/\|d_k\|$. This selection was used for the first time by Shanno and Phua in CONMIN [29]. It was also considered in the packages: SCG by Birgin and Martínez [9] and in SCALCG by Andrei [3-6]. In step 7 we use $\delta = 0$ and only when $\|s_k\| \leq \tau$, where τ is a small specified constant, is the modified secant condition (2.2) considered, i.e. we set $\delta = 1$ in our numerical experiments.

Proposition 5.1. *Suppose that f is a uniformly convex function on the level set $S = \{x : f(x) \leq f(x_0)\}$, and d_k satisfies the sufficient descent condition $g_k^T d_k < -c_1 \|g_k\|^2$, where $c_1 > 0$, and $\|d_k\|^2 \leq c_2 \|g_k\|^2$, where $c_2 > 0$. Then the sequence generated by ACGMSEC converges linearly to x^*, solution to the problem (1.1).*

Proof. From (4.12) we have that $f(x_{k+1}) \leq f(x_k)$ for all k. Since f is bounded below, it follows that

$$\lim_{k\to\infty}(f(x_k) - f(x_{k+1})) = 0.$$

Now, since f is uniformly convex there exist positive constants m and M, such that $mI \leq \nabla^2 f(x) \leq MI$ on S. Suppose that $x_k + \alpha d_k \in S$ and $x_k + \gamma_m \alpha d_k \in S$ for all $\alpha > 0$. We have:

$$f(x_k + \gamma_m \alpha d_k) \leq f(x_k + \alpha d_k) - \frac{(a_k + b_k)^2}{2b_k}.$$

But, from uniform convexity we have the following quadratic upper bound on $f(x_k + \alpha d_k)$:

$$f(x_k + \alpha d_k) \leq f(x_k) + \alpha g_k^T d_k + \frac{1}{2} M \alpha^2 \|d_k\|^2.$$

Therefore,

$$\begin{aligned} f(x_k + \alpha d_k) &\leq f(x_k) - \alpha c_1 \|g_k\|^2 + \frac{1}{2} M c_2 \alpha^2 \|g_k\|^2 \\ &= f(x_k) + \left[-c_1 \alpha + \frac{1}{2} M c_2 \alpha^2 \right] \|g_k\|^2. \end{aligned}$$

Observe that for $0 \leq \alpha \leq c_1/(Mc_2)$, $-c_1\alpha + \frac{1}{2} M c_2 \alpha^2 \leq -\frac{c_1}{2}\alpha$ which follows from the convexity of $-c_1\alpha + (Mc_2/2)\alpha^2$. Using this result we get:

$$f(x_k + \alpha d_k) \leq f(x_k) - \frac{1}{2} c_1 \alpha \|g_k\|^2 \leq f(x_k) - \rho c_1 \alpha \|g_k\|^2,$$

since $\rho < 1/2$.

From proposition 4.1 the Wolfe line search terminates with a value $\alpha \geq \omega > 0$. Therefore, for $0 \leq \alpha \leq c_1/(Mc_2)$, this provides a lower bound on the decrease in the function f, i.e.

$$f(x_k + \alpha d_k) \leq f(x_k) - \rho c_1 \omega \|g_k\|^2. \tag{5.1}$$

On the other hand,

$$\frac{(a_k + b_k)^2}{2b_k} \geq \frac{(\alpha^2 M c_2 - \alpha c_1)^2 \|g_k\|^4}{2\alpha^2 M c_2 \|g_k\|^2} \geq \frac{(\omega M c_2 - c_1)^2}{2Mc_2} \|g_k\|^2. \tag{5.2}$$

Considering (5.1) and (5.2) we get:

$$f(x_k+\gamma_m\alpha d_k)\le f(x_k)-\rho c_1\omega\|g_k\|^2-\frac{(\omega Mc_2-c_1)^2}{2Mc_2}\|g_k\|^2. \tag{5.3}$$

Therefore,

$$f(x_k)-f(x_k+\gamma_m\alpha d_k)\ge\left[\rho c_1\omega+\frac{(\omega Mc_2-c_1)^2}{2Mc_2}\right]\|g_k\|^2.$$

But, $f(x_k)-f(x_{k+1})\to 0$ and as a consequence g_k goes to zero, i.e. x_k converges to x^*. Having in view that $f(x_k)$ is a nonincreasing sequence, it follows that $f(x_k)$ converges to $f(x^*)$. From (5.3) we see that

$$f(x_{k+1})\le f(x_k)-\left[\rho c_1\omega+\frac{(\omega Mc_2-c_1)^2}{2Mc_2}\right]\|g_k\|^2. \tag{5.4}$$

Combining this with $\|g_k\|^2\ge 2m(f(x_k)-f^*)$ and subtracting f^* from both sides of (5.4) we conclude:

$$f(x_{k+1})-f^*\le c(f(x_k)-f^*),$$

where

$$c=1-2m\left[\rho c_1\omega+\frac{(\omega Mc_2-c_1)^2}{2Mc_2}\right]<1.$$

Therefore, $f(x_k)$ converges to f^* at least as fast as a geometric series with a factor that depends on the parameter ρ in the first Wolfe condition and the bounds m and M. So, the convergence of the acceleration scheme is at least linear. ■

6. Numerical results and comparisons

In this section we report some numerical results obtained with a Fortran implementation of the ACGMSEC algorithm. The code is written in Fortran and compiled with f77 (default compiler settings) on a Workstation Intel Pentium 4 with 1.8 GHz. We selected a number of 75 large-scale unconstrained optimization test functions (some from CUTE library [10]) in generalized or extended form [1]. For each test function we have taken ten numerical experiments with the number of variables $n=1000,2000,...,10000$. The algorithm implements the Wolfe line search conditions with $\rho=0.0001$ and $\sigma=0.9$, and also the same stopping criterion $\|g_k\|_\infty\le 10^{-6}$, where $\|.\|_\infty$ is the maximum absolute component of a vector. The comparisons of algorithms are given in the following context. Let f_i^{ALG1} and f_i^{ALG2} be the optimal value 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:

$$\left|f_i^{ALG1}-f_i^{ALG2}\right|<10^{-3} \tag{6.1}$$

and the number of iterations, or the number of function-gradient evaluations, 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 ACGMSEC with $\tau=0$ versus some conjugate gradient algorithms. Figures 1-6 present the Dolan and Moré [15] CPU performance profile of ACGMSEC versus Hestenes-Stiefel [21], Polak-Ribière-Polyak [25, 26], Dai-Yuan [12], Dai-Liao [11], hybrid Dai-Yuan and hybrid Dai-Yuan zero [12], respectively.

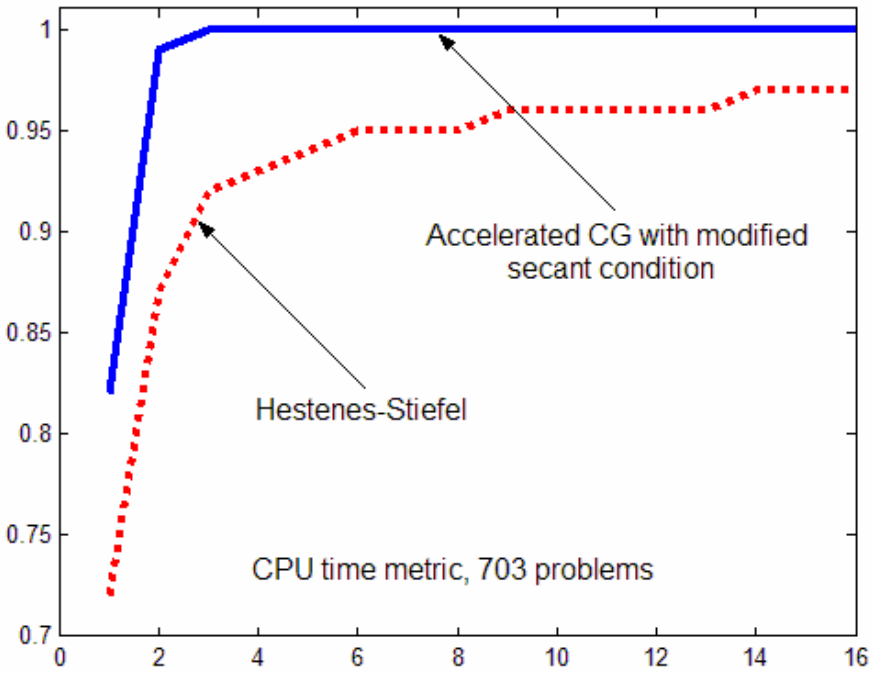

Fig. 1. ACGMSEC ($\tau = 0$) versus Hestenes-Stiefel.

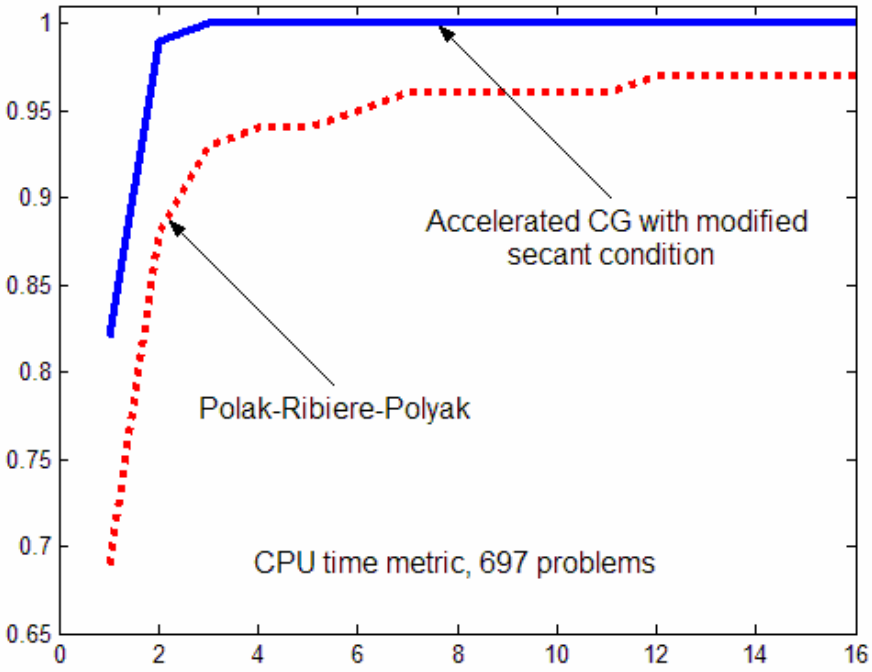

Fig. 2. ACGMSEC ($\tau = 0$) versus Polak-Ribière-Polyak.

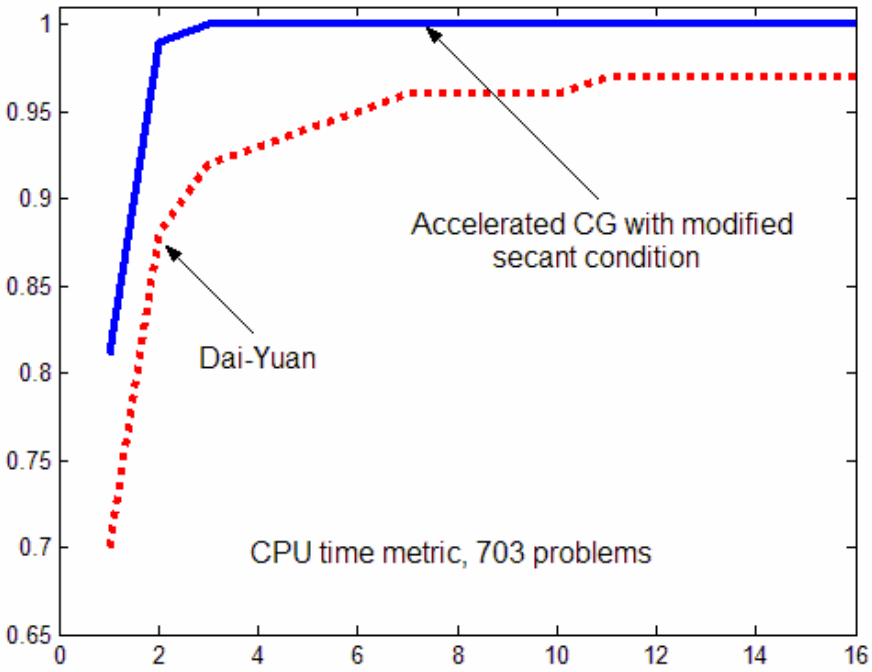

Fig. 3. ACGMSEC ($\tau = 0$) versus Dai-Yuan.

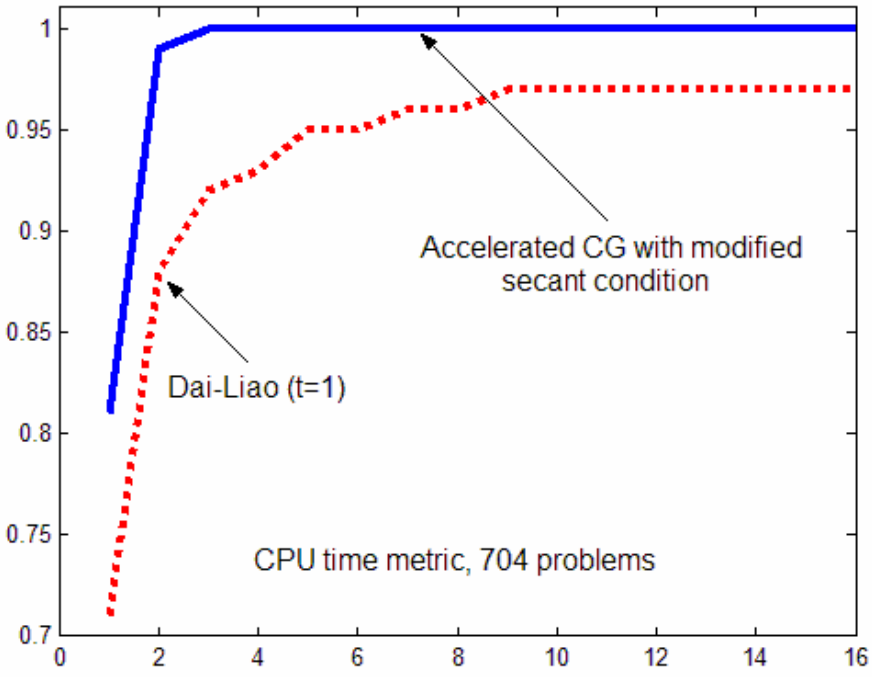

Fig. 4. ACGMSEC ($\tau = 0$) versus Dai-Liao (t=1).

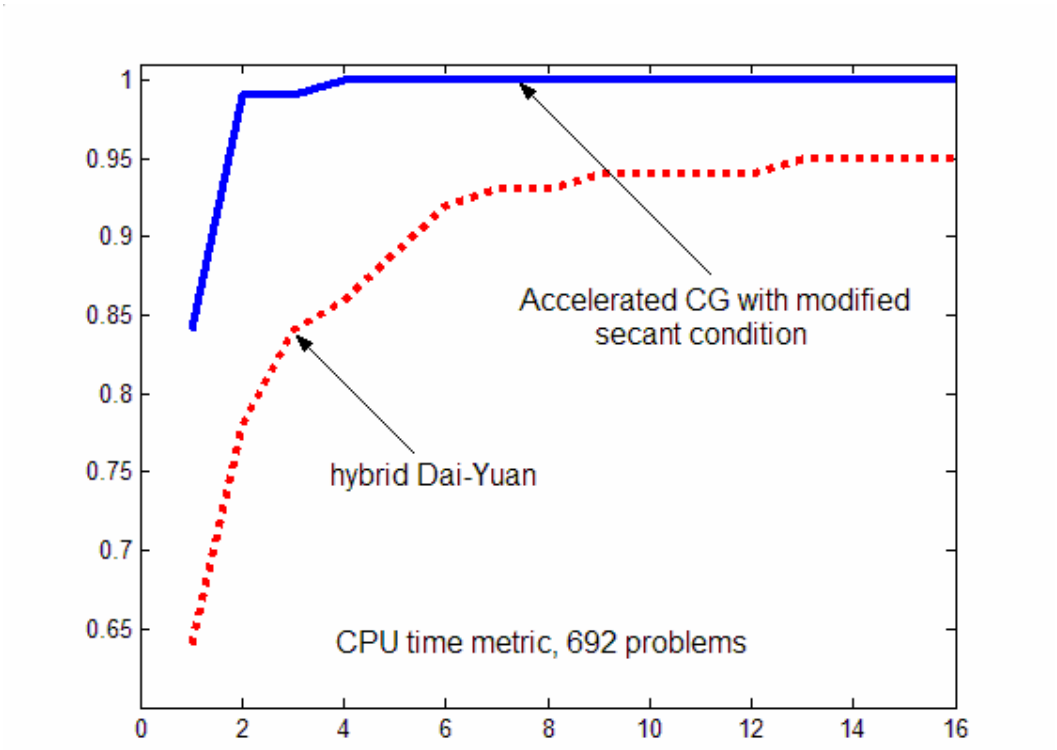

Fig. 5. ACGMSEC ($\tau = 0$) versus hybrid Dai-Yuan.

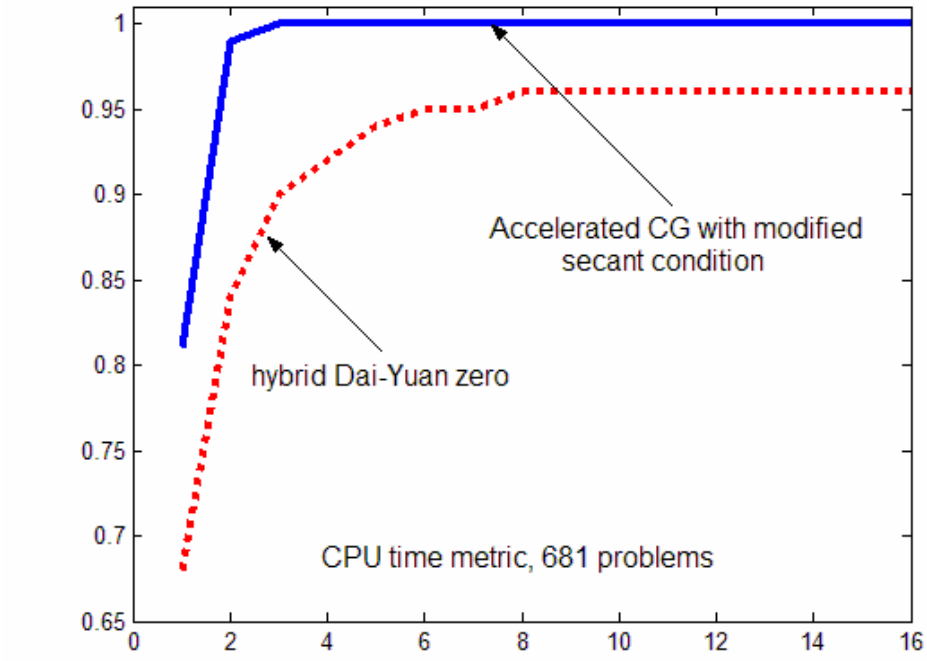

Fig. 6. ACGMSEC ($\tau = 0$) versus hybrid Dai-Yuan zero.

When comparing ACGMSEC with all these conjugate gradient algorithms subject to CPU time metric we see that ACGMSEC is top performer, i.e. the accelerated conjugate gradient algorithm with modified secant condition is more successful and more robust than the considered conjugate gradient algorithms. 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.

In the second set of numerical experiments we compare ACGMSEC with CG_DESCENT by Hager and Zhang [19]. Figures 7 and 8 present the Dolan and Moré CPU performance profiles of ACGMSEC versus these algorithms.

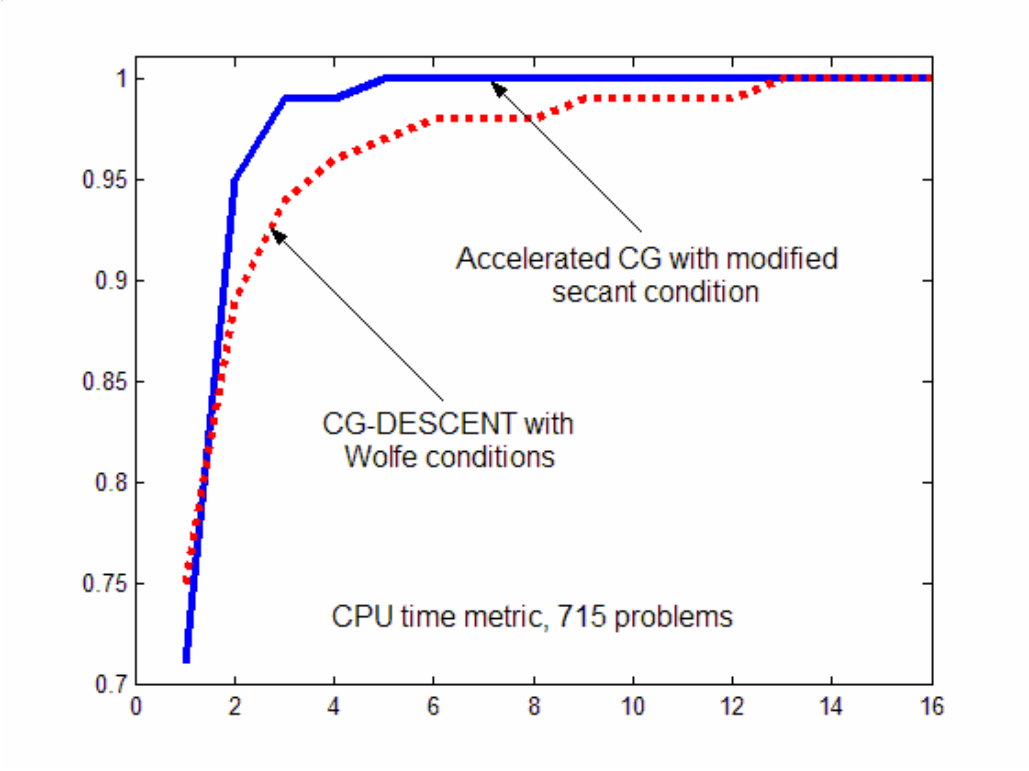

Fig. 7. ACGMSEC ($\tau = 0$) versus CG_DESCENT with Wolfe conditions.

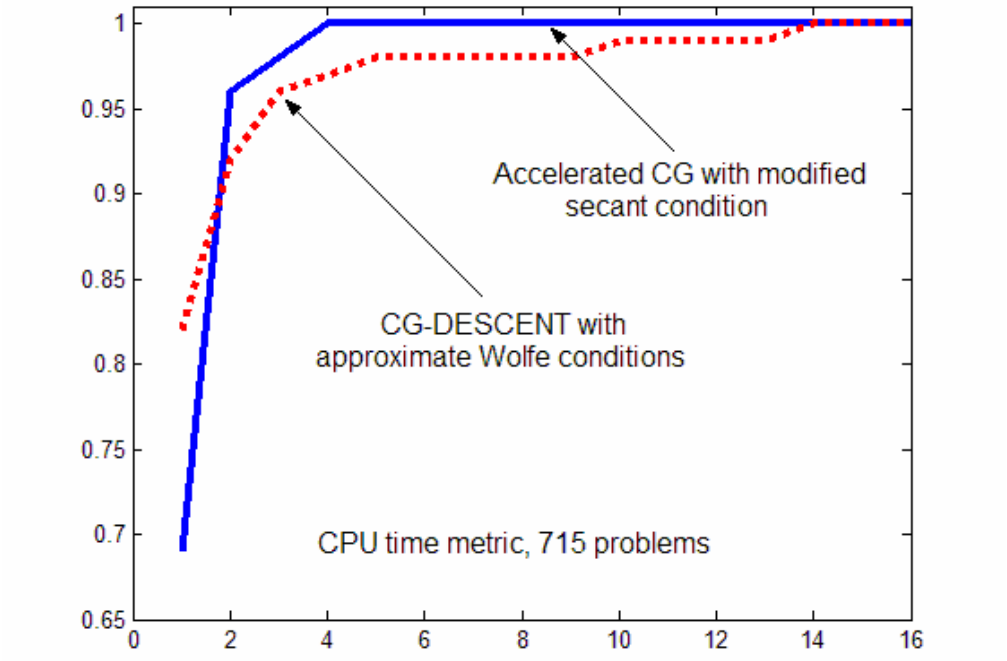

Fig. 8. ACGMSEC ($\tau = 0$) versus CG_DESCENT with approximate Wolfe conditions.

From the Figures above we see that ACGMSEC is again the top performer. The modified secant condition and acceleration scheme are very important ingredients in designing efficient conjugate gradient algorithms.

In the third set of numerical experiments we compare ACGMSEC with the CONMIN conjugate gradient algorithm by Shanno and Phua [29]. Figure 9 presents the Dolan and Moré CPU performance profiles of ACGMSEC versus CONMIN.

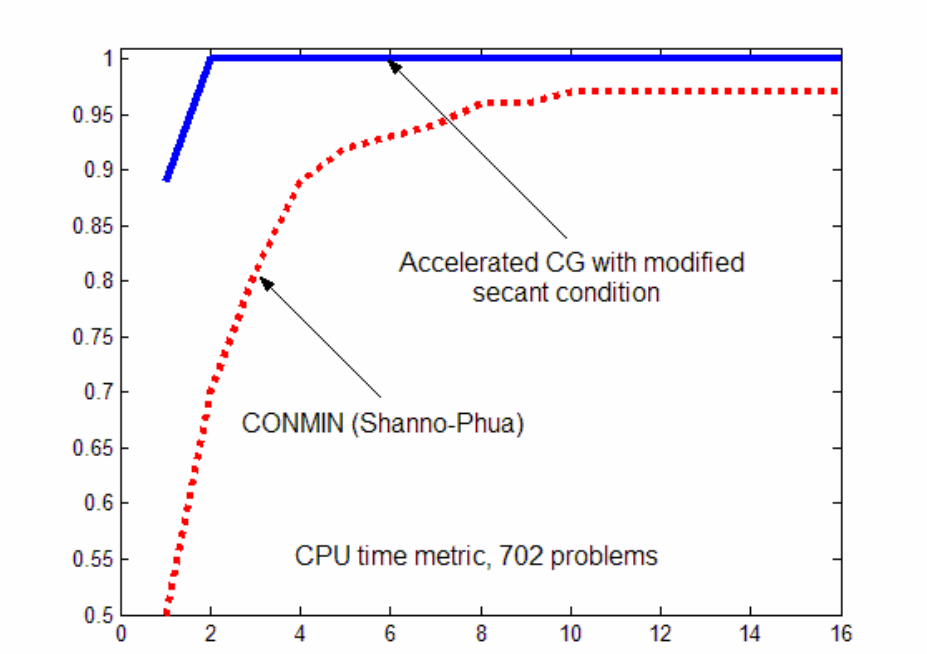

Fig. 9. ACGMSEC ($\tau = 0$) versus CONMIN (Shanno-Phua).

In the fourth set of numerical experiments we compare ACGMSEC with SCALCG by Andrei [3-5]. Figure 10 presents the Dolan and Moré CPU performance profiles of ACGMSEC versus SCALCG (spectral).

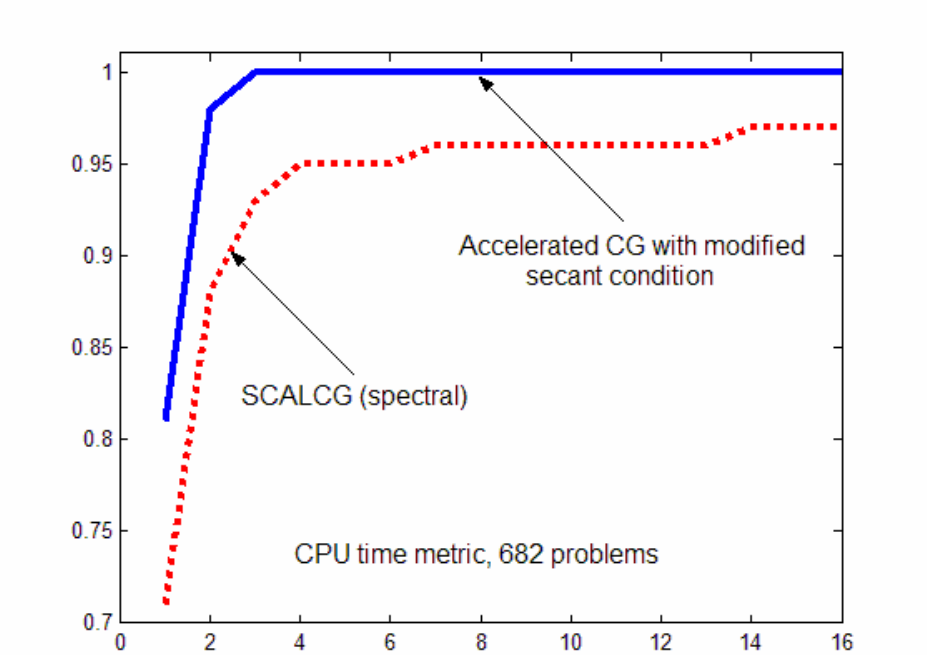

Fig. 10. ACGMSEC ($\tau = 0$) versus SCALCG (spectral).

Finally, in Figure 11 we present a comparison between ACGMSEC and LBFGS (m=3) by Liu and Nocedal [22].

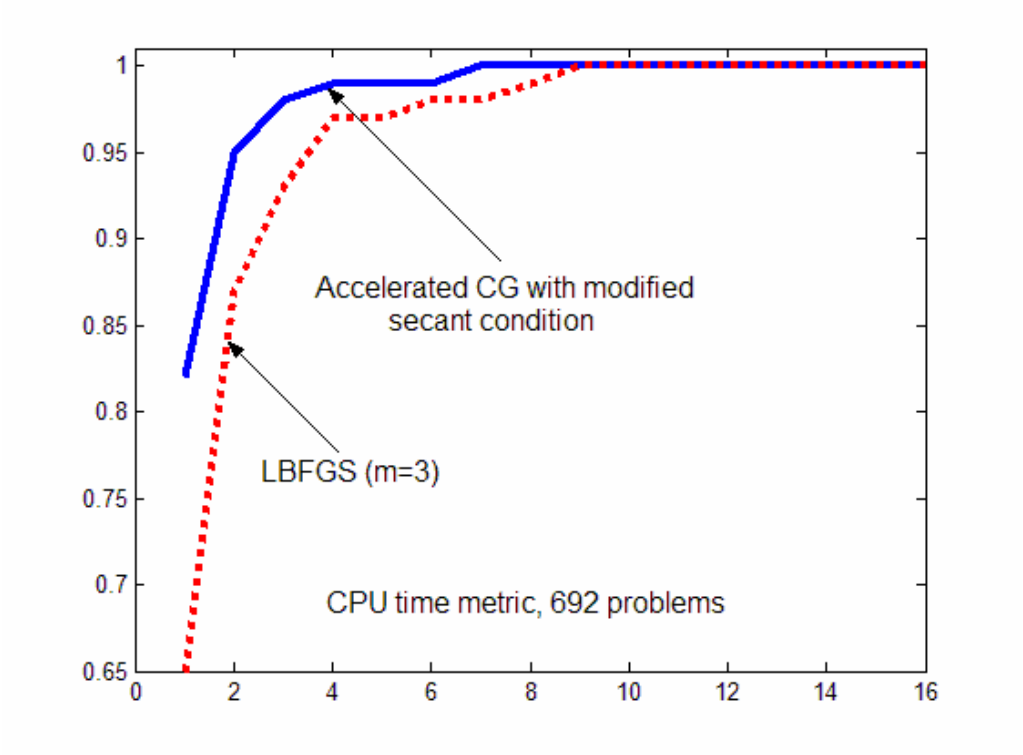

Fig. 11. ACGMSEC ($\tau = 0$) versus LBFGS (m=3).

From Figures 7 and 8 we see that the performances of ACGMSEC and CG_DESCENT are comparable, ACGMSEC being slightly faster. CG_DESCENT is a highly elaborated modification of the Hestenes and Stiefel method. This modification is scale invariant, it goes to zero when the iterates jamm and it enhances descent. On the other hand, ACGMSEC uses the second order information in a very clever manner through the modified secant condition. Although we are not able to prove the global convergence for (1.2), (1.3) and (2.6), we established the global convergence for general nonlinear functions by restricting β_k like in (3.16). At present this is a classical approach. Similar results have been obtained for the Polak-Ribière-Polyak, Fletcher-Revees, Dai-Yuan and CG_DESCENT versions of the conjugate gradient methods. The novelty of the ACGMSEC algorithm is given by the incorporation of the second order information through the modified secant condition and through the acceleration scheme.

7. Conclusion

We have presented a new conjugate gradient algorithm (ACGMSEC) for solving large-scale unconstrained optimization problems. The algorithm exploits the presence of the Hessian in the formula for β_k computation as well as the fact that the step lengths in conjugate gradient algorithms differ from 1 in the vast majority of iterations. The algorithm approximates the Hessian / vector product by means of the modified secant condition. It modifies the step length by an acceleration scheme which proved to be very efficient in reducing the values of the minimizing function along the iterations. We proved that if the direction is a descent one, then the algorithm is globally convergent. For uniformly convex functions the convergence of the accelerated scheme is still linear, but the reduction in function values is significantly improved. For a test set consisting of 750 problems (some of them from CUTE library) with dimensions ranging between 1000 and 10,000, the CPU time performance profiles of ACGMSEC was higher than those of HS, PRP, DY, DL (t=1), hDY, hDYz, CG_DESCENT,

CONMIN, SCALCG and LBFGS (m=3). Both the above ingredients based on the modified secant condition and on the acceleration scheme are crucial for the efficiency of the algorithm.

References

1. N. Andrei, "*Test functions for unconstrained optimization*". http://www.ici.ro/camo/neculai/HYBRID/evalfg.for
2. N. Andrei, *An acceleration of gradient descent algorithm with backtracking for unconstrained optimization*, Numerical Algorithms, Vol. 42, pp. 63-73, 2006.
3. N. Andrei, *Scaled conjugate gradient algorithms for unconstrained optimization*. Computational Optimization and Applications, 38 (2007), pp. 401-416.
4. N. Andrei, *Scaled memoryless BFGS preconditioned conjugate gradient algorithm for unconstrained optimization.* Optimization Methods and Software, 22 (2007), 561-571.
5. N. Andrei, *A scaled BFGS preconditioned conjugate gradient algorithm for unconstrained optimization*. Applied Mathematics Letters, 20 (2007), 645-650.
6. N. Andrei, *Numerical comparison of conjugate gradient algorithms for unconstrained optimization.* Studies in Informatics and Control, 16 (2007), pp.333-352.
7. N. Andrei, *Acceleration of conjugate gradient algorithms for unconstrained optimization.* Submitted.
8. N. Andrei, *Accelerated conjugate gradient algorithm with Hessian / vector product approximation for unconstrained optimization.* ICI Technical Report, February 2008.
9. E. Birgin and J.M. Martínez, *A spectral conjugate gradient method for unconstrained optimization*, Applied Math. and Optimization, 43, pp.117-128, 2001.
10. I. Bongartz, A.R. Conn, N.I.M. Gould and P.L. Toint, *CUTE: constrained and unconstrained testing environments*, ACM Trans. Math. Software, 21, pp.123-160, 1995.
11. Y.H. Dai and L.Z. Liao, *New conjugacy conditions and related nonlinear conjugate gradient methods.* Appl. Math. Optim., 43 (2001), pp. 87-101.
12. Y.H. Dai and Y. Yuan, *An efficient hybrid conjugate gradient method for unconstrained optimization*, Ann. Oper. Res., 103 (2001), pp. 33-47.
13. Y.H. Dai, Han, J.Y., Liu, G.H., Sun, D.F., Yin, .X. and Yuan, Y., *Convergence properties of nonlinear conjugate gradient methods*. SIAM Journal on Optimization 10 (1999), 348-358.
14. J.W. Daniel, *The conjugate gradient method for linear and nonlinear operator equations.* SIAM J. Numer. Anal., 4 (1967), pp.10-26.
15. E.D. Dolan and J.J. Moré, *Benchmarking optimization software with performance profiles*, Math. Programming, 91 (2002), pp. 201-213.
16. J.C. Gilbert and J. Nocedal, *Global convergence properties of conjugate gradient methods for optimization*, SIAM J. Optim., 2 (1992), pp. 21-42.
17. A.A. Goldstein, *On steepest descent,* SIAM J. Control, Vol. 3, pp.147-151, 1965.
18. A. Griewank and Ph.L. Toint, *On the unconstrained optimization of partially separable objective functions.* in Nonlinear Optimization 1981, (M.J.D. Powell Ed.), Academic Press, London, pp.301-312.
19. 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) 170-192.
20. W.W. Hager and H. Zhang, *A survey of nonlinear conjugate gradient methods*. Pacific journal of Optimization, 2 (2006), pp.35-58.
21. M.R. Hestenes and E.L. Stiefel, *Methods of conjugate gradients for solving linear systems*, J. Research Nat. Bur. Standards, 49 (1952), pp.409-436.
22. D.C. Liu and J. Nocedal, *On the limited memory BFGS method for large scale optimization methods.* Mathematical Programming, 45 (1989), pp. 503-528.
23. J. Nocedal, *Conjugate gradient methods and nonlinear optimization*. In Linear and nonlinear Conjugate Gradient related methods, L. Adams and J.L. Nazareth (eds.), SIAM, 1996, pp.9-23.

24. A. Perry, *A modified conjugate gradient algorithm.* Discussion Paper no. 229, Center for Mathematical Studies in Economics and Management Science, Northwestern University, (1976).
25. E. Polak and G. Ribière, *Note sur la convergence de directions conjuguée,* Rev. Francaise Informat Recherche Operationelle, 3e Année 16 (1969), pp.35-43.
26. B.T. Polyak, *The conjugate gradient method in extreme problems*. USSR Comp. Math. Math. Phys., 9 (1969), pp.94-112.
27. T. Schlick, and A. Fogelson, *TNPACK - A truncated Newton minimization package for large-scale problems: I Algorithm and usage.* ACM Transactions on Mathematical Software, 18 (1992), pp. 46-70.
28. T. Schlick, and A. Fogelson, *TNPACK - A truncated Newton minimization package for large-scale problems: II Implementation examples.* ACM Transactions on Mathematical Software, 18 (1992), pp. 71-111.
29. D.F. Shanno and K.H. Phua, *Algorithm 500, Minimization of unconstrained multivariate functions*, ACM Trans. on Math. Soft., 2, pp.87-94, 1976.
30. J.Z. Zhang, N.Y. Deng and L.H. Chen, *New quasi-Newton equation and related methods for unconstrained optimization.* J. Optim. Theory Appl., 102 (1999), pp.147-167.
31. J.Z. Zhang, C.X. Xu, *Properties and numerical performance of quasi-Newton methods with modified quasi-Newton equations.* Journal of Computational and Applied Mathematics, 137 (2001), pp.269-278.
32. H. Yabe and M. Takano, *Global convergence properties of nonlinear conjugate gradient methods with modified secant condition*. Computational Optimization and Applications, 28 (2004), pp.203-225.

March 3, 2008