International Journal of Mathematical Archive-3(5), 2012, 2035-2046
IJMA Available online through www.ijma.info ISSN 2229 – 5046

COMPUTATIONAL TREATMENTS OF AN IMPROVED CONJUGATE GRADIENT METHOD FOR UNCONSTRAINED MINIMIZATION

***Abbas Y. Al-Bayati & **Walied H. Sharif**

**College of Basic Education Telafer, University of Mosul, Mosul, Iraq*
***Department of Mathematics and Computer, Foundation Program of Qatar, University of Qatar, Doha, Qatar*

(Received on: 05-05-12; Accepted on: 27-05-12)

ABSTRACT

R*ecently, the authors introduced a new three-terms nonlinear Conjugate Gradient (CG) method [2] for solving unconstrained optimization problems. Their method was compared with the well-known Zhang's three-terms CG-method [32]. This paper contains a description of several new restarting procedures for the same proposed CG-method introduced by the authors and a numerical investigation of the influence of several scaling techniques with a modified perfect cubic line search procedure on their efficiency. Computational results obtained by means of (35) sufficiently difficult problems are given with promising numerical results.*

Key Words*: Conjugate Gradient Method, Unconstrained Optimization, Convergence Property, Line Searches, Restarting and Scaling Techniques, Large-Scale Problems, Computational Numerical Experiments.*

AMS (2010) Subject Classification*: 90C06, 90C26.*

1. INTRODUCTION:

In this study, we are concerned with the minimization of an unconstrained optimization problem to find a local minimum $x^* \in R^n$ of the function $f : X \to R$ on an open set $X \subset R^n$; i.e. a point $x^* \in R^n$ that satisfies the inequality $f(x^*) \le f(x) \;\; \forall\, x \in B(x^*, \varepsilon)$ for some $\varepsilon > 0$, where $B(x^*, \varepsilon) = \{x \in R^n : \|x - x^*\| < \varepsilon\} \subset X$ is an open ball contained in $X \subset R^n$; in other words we want to:

$$\min\{f(x) \mid x \in R^n\} \tag{1}$$

where $f : R^n \to R$ is a continuously differentiable function, and its gradient at point x_k is denoted by g_k for the sake of simplicity. n is the number of variables, which is automatically assumed to be large. The iterative formula of nonlinear CG-method is given by:

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

where α_k is a step-length, and d_k is a search direction which is determined by:

$$d_k = \begin{cases} -g_0, & \text{if } k = 0, \\ -g_k + \beta_k d_{k-1}, & \text{if } k \ge 1, \end{cases} \tag{3}$$

where β_k is a scalar and d_k is a direction vector satisfying the equation $Bd_k + g_k = 0$, where B is a symmetric positive definite approximation of the Hessian matrix that is constructed iteratively [18]. If the number of variables is large, then matrix B cannot be stored, nor factored in a reasonable time, so other methods have to be used. There exist several classes of such methods: Conjugate Gradient (CG) methods [9], difference versions of Truncated Newton (TN) methods [8], Variable Metric (VM) methods with limited storage [20], sparse variants of VM-methods [27], and partitioned VM-methods for separable problems [12]. The last two classes require the special structure of optimization

Corresponding author: **Walied H. Sharif
***Department of Mathematics and Computer, Foundation Program of Qatar,University of Qatar, Doha, Qatar*

problems. From the other classes the simplest are the CG-methods which need only 3-5 n-dimensional vectors (it depends on their implementation). Recently new attention has been given to these methods because they are globally convergent with mild and reasonable assumptions. Their idea starts since 1952, there have been many well-known formulas for the scalar β_k, for example, Fletcher-Reeves (FR), Polak-Ribiere (PR), Hestenes-Stiefel (HS) and Dai-Yuan (DY) [6, 21]:

$$\beta_k^{FR} = \frac{\|g_k\|^2}{\|g_{k-1}\|^2}, \quad \beta_k^{PR} = \frac{g_k^T y_{k-1}}{\|g_{k-1}\|^2}, \quad \beta_k^{HS} = \frac{g_k^T y_{k-1}}{d_{k-1}^T y_{k-1}}, \quad \beta_k^{DY} = \frac{\|g_k\|^2}{d_{k-1}^T y_{k-1}}, \tag{4}$$

where $y_{k-1} = g_k - g_{k-1}$, symbol $\|\,.\,\|$ denotes the Euclidean norm of vectors. If f is a strictly convex quadratic function, all these methods are equivalent in the case that an Exact Line Search (ELS) is used. If the objective function is non-convex, their behaviors may be distinctly different. In the past two decades, the convergence properties CG-methods defined in (4) have been intensively studied by many researchers [1, 5, 11, 13, 14, 16, 26, 29, 33]. Although the HS method is most general and the FR method is the simplest with good global convergence properties (theoretical), the most numerically efficient was proved to be the PR method. Another important issue related to the performance of CG-methods is the line search, which requires sufficient accuracy to ensure that the search directions yield descent [15]. Common criteria for line search accuracy are the Wolfe conditions [30, 31]:

$$f(x_{k-1} + \alpha_k d_k) - f(x_{k-1}) \le -\delta \alpha_k g_{k-1}^T d_{k-1}, \tag{5a}$$

$$g_k^T d_{k-1} \ge \sigma g_{k-1}^T d_{k-1}, \tag{5b}$$

where $0 < \delta \le \sigma < 1$. In the "Strong Wolfe" conditions, (5b) is replaced by

$$\left| g_k^T d_{k-1} \right| \le -\sigma g_{k-1}^T d_{k-1}\,. \tag{5c}$$

It has been shown [7] that for the FR scheme, the strong Wolfe conditions may not yield a direction of descent unless $\sigma \le 1/2$. However, The CG-methods are more sensitive to their implementation than the VM methods:

1. The initial estimate α_1 of α_k in the line search algorithm does not have theoretical justification for CG-methods. Therefore the CG-methods are more sensitive to the initial estimate α_1 than the VM methods.
2. CG-methods need more perfect line search than VM methods. We usually use $\delta = 0.1$ for CG- VM-methods.
3. CG-methods strongly depend on restarts while VM-methods need not be restarted.

2. PRELIMINARIES:

2.1. Assumption: The objective function *f* is bounded below, and the level set

$$F = \left\{ x \in R^n \,\middle|\, f(x) \le f(x_0) \right\} \text{ is bounded.} \tag{6}$$

2.2. Assumption: In some neighborhood *N* of *F*, f is differentiable and its gradient is Lipschitz continuous, namely, there exists a positive constant *L* such that:

$$\|g(x) - g(y)\| \le L\|x - y\|, \quad \forall x, y \in N \tag{7a}$$

The above assumption implies that there exists a positive constant $\bar{\gamma}$ such that

$$\|g(x)\| \le \bar{\gamma}, \quad \forall x \in F \tag{7b}$$

2.3. Zhang's Three-Terms CG-Method [32]:

Zhang, et al. had introduced a three-term CG method as follows:

$$d_k = \begin{cases} -g_0, & if \;\; k = 0, \\ -g_k + \beta_k^{DL} d_{k-1} - \xi_k \left(y_{k-1} - t s_{k-1} \right), & if \;\; k \ge 1, \end{cases} \tag{8a}$$

where

$$t \geq 0; \qquad \xi_k = \frac{g_k^T d_{k-1}}{d_{k-1}^T y_{k-1}} \text{ and } \beta_k^{DL} = \frac{g_k^T (y_{k-1} - t s_{k-1})}{d_{k-1}^T y_{k-1}}, \tag{8b}$$

They show that the sufficient descent condition also holds true if no line search is used, that is,

$$g_{k-1}^T d_k = -\|g_{k-1}\|^2. \tag{9}$$

In order to achieve the global convergence result, Grippo and Lucidi [13] proposed a new line search. For given constants $\tau > 0$, $\delta > 0$, and $\lambda \in (0,1)$, let

$$\alpha_k = \max\left\{ \lambda^j \left(\frac{\tau |g_k^T d_k|}{\|d_k\|^2} \right); \quad j = 0, 1, \ldots \right\} \tag{10a}$$

satisfy

$$f(x_k) \leq f(x_{k-1}) - \delta \alpha_k^2 \|d_{k-1}\|^2 \tag{10b}$$

This line search will be preferred to the classical Armijo one for the sake of a greater reduction of objective function. Introducing this line search rule, we are now ready to state the outline of the Zhang, et al. [32] first three-term CG-method as follows:

2.4. Outline of Zhang's Three-Terms CG-Algorithm [32]:

Step 1. Given $x_0 \in R^n$. Let $0 < \delta < \sigma < 1$, $t \geq 0$ and $d_0 = -g_0$. Set $k := 0$.

Step 2. If $\|g_k\| \leq 10^{-6}$, then stop.

Step 3. Compute d_k using (8).

Step 4. Find the step-length α_k satisfying (11) and (12).

$$f(x_{k-1} + \alpha_k d_k) - f(x_{k-1}) \leq -\delta \alpha_k^2 \|d_{k-1}\|^2, \tag{11}$$

$$g(x_{k-1} + \alpha_k d_k)^T d_k \geq \sigma g_{k-1}^T d_{k-1}. \tag{12a}$$

$$|g_k^T d_{k-1}| \leq -\sigma g_{k-1}^T d_{k-1} \tag{12b}$$

and set $x_k = x_{k-1} + \alpha_k d_k$.

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

2.5. Al-Bayati and Altae Three-Term CG-Method [2]:

The search directions of this method are defined by; see Al-Bayati and Altae:

$$d_k = \begin{cases} -g_0, & if \quad k = 0, \\ -g_k + \beta_k^N d_{k-1} - \xi_k \left(y_{k-1} - \left(2 \frac{\|y_{k-1}\|^2}{s_{k-1}^T y_{k-1}}\right) s_{k-1} \right), & if \quad k \geq 1, \end{cases} \tag{13a}$$

where

$$\beta_k^N = \max\left\{ \frac{g_k^T y_{k-1}}{d_{k-1}^T y_{k-1}}, 0 \right\} - t_k \frac{g_k^T s_{k-1}}{d_{k-1}^T y_{k-1}}, \tag{13b}$$

and

$$\xi_k = \frac{g_k^T d_{k-1}}{d_{k-1}^T y_{k-1}}; \qquad t_k = 2\frac{\|y_{k-1}\|^2}{s_{k-1}^T y_{k-1}} \geq 0 \tag{13c}$$

It is easy to see that the sufficient descent condition (9) also holds true if no line search is used.

2.6. Outline of Al-Bayati and Altae Three-Term CG-Algorithm [2]:

Step 1. Given $x_0 \in R^n$. Let $0 < \delta < \sigma < 1$, $t \geq 0$ and $d_0 = -g_0$. Set $k := 0$.

Step 2. If $\|g_k\| \leq 10^{-6}$, then stop.

Step 3. Compute d_k using (13).

Step 4. Find the step-length α_k satisfying (11) and (12) and set $x_k = x_{k-1} + \alpha_k d_k$.

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

3. MODIFIED RESTARTING; SCALING AND LINE SEARCH TECHNIQUES IMPLEMENTED IN THE NEW PROPOSED CG-ALGORITHM:

In this section we are going to introduce several new restarting and several scaling techniques to improve the performance behavior of Al-Bayati and Altae [2] three-terms CG-algorithm and as follows:

3.1. Different Restarting Techniques:

We limit our attention to the PR method, but the same considerations can be used for the HS method. Usually the PR method is implemented with periodic restarts. In [23], Powell points out that the PR method works better if it is restarted whenever

$$\beta_k^{PR} < 0. \tag{14}$$

The PR method with periodic restarts can be disadvantageous for some problems that require more restarts at the beginning of the iterative process. Also, convergence results noted in this paper that our computational experiments show that the PR method is more efficient if it is restarted not only when (14) holds, but also whenever:

$$\beta_k^{PR} \leq \eta \beta_k^{FR} \tag{15}$$

and

$$\lambda \|g_k\|^2 \leq \omega \tag{16}$$

where $1 < \eta < 1/(2\sigma)$ is a suitable constant (we recommend $\lambda = 10^{-8}$, $\eta = 1.34$, $\omega = 10^{-4}$ all recommended values given in this paper were obtained experimentally by means of extensive computations).

3.2. Different Scaling Techniques:

Another useful tool for improving CG methods is scaling, which was originally developed for VM methods [24]. The scaling consists in replacing (3) by:

$$d_k = \gamma_k \left(-g_k + \beta_k d_{k-1} - t_k s_{k-1}\right), \tag{17}$$

where γ_k is the scaling factor. This type of CG-methods are called spectral CG-methods. Then it found that the best value of this parameter:

$$\gamma_k = \frac{y_{k-1}^T s_{k-1}}{y_{k-1}^T y_{k-1}}, \tag{18}$$

Note that when we use (17) then PR; FR and DY have to be replaced by:

$$\beta_{PR} = \frac{1}{\gamma_k}\left(\frac{y_{k-1}^T g_k}{g_{k-1}^T g_{k-1}}\right), \tag{19a}$$

and

$$\beta_{FR} = \frac{1}{\gamma_k}\left(\frac{g_k^T g_k}{g_{k-1}^T g_{k-1}}\right), \tag{19b}$$

and

$$\beta_{DY} = \frac{1}{\gamma_k}\left(\frac{g_k^T g_k}{d_{k-1}^T y_{k-1}}\right) \tag{19c}$$

For the simplification of subsequent considerations, we have used the following scaling criterion to scale our new proposed three-terms CG-algorithm.

$$\gamma_k = \bar{\gamma}_1, \quad \text{if} \quad \frac{y_{k-1}^T s_{k-1}}{y_{k-1}^T y_{k-1}} < \bar{\gamma}_1 \tag{20a}$$

$$\gamma_k = \bar{\gamma}_2, \quad \text{if} \quad \frac{y_{k-1}^T s_{k-1}}{y_{k-1}^T y_{k-1}} < \bar{\gamma}_2 \tag{20b}$$

$$\gamma_k = \frac{y_{k-1}^T s_{k-1}}{y_{k-1}^T y_{k-1}}, \text{ otherwise,} \tag{20c}$$

where $0 < \bar{\gamma}_1 < 1 < \bar{\gamma}_2$ (we recommend $\bar{\gamma}_1 = 0.005$ and $\bar{\gamma}_2 = 200$). The bounds $\bar{\gamma}_1$ and $\bar{\gamma}_2$ serve for improvement of stability.

3.3. Perfect Cubic Line Search Technique:

Since the CG-methods require more **perfect line search** than other methods, they are very sensitive to its realization. We have essentially used the standard cubic line search implementation; namely (a perfect cubic interpolation), which can be represented by the following algorithm:

3.4. Line Search Algorithm: Input data: $\Delta > 0$; $0 < \varphi_1 < \varphi_2 < 1$ and $0 < \delta < \sigma < 1/2$.
$\delta = 0.0001; \sigma = 0.1$, $\varphi_1 = 0.01$, $\varphi_2 = 0.9$ and $\Delta = 1000$

Step (1): Determine the initial estimate α_1 of α_k. This is may be taken as: $\alpha_1 = (1, \ \textit{or} \ \min\left(1,\, 2\frac{f_{\min} - f_1}{s_1^T g_1}\right))$

and set $(\psi_1 = 0,\ i = 1)$

Step (2): Set $\alpha_i = \min(\alpha_i,\, \Delta/\|s_k\|)$. Set $\rho_i = \psi_i$ and $\psi_i = \alpha_i$. If the conditions (5a) and (5c) are satisfied with f_{k+1} and g_{k+1} replaced by $f(x_k + \alpha_i s_k)$ and $g(x_k + \alpha_i s_k)$ respectively, then set $\alpha_k = \alpha_i$ and **terminate the computation**. If both (5a) and $s_k^T g(x_k + \alpha_i s_k) < 0$, hold then go to Step(3), else go to Step(4).

Step (3): If $\alpha_i = \Delta/\|s_k\|$ then set $\alpha_k = \alpha_i$ and **terminate the computation**, else determine the new estimate α_i by **cubic extrapolation**. Set $\alpha_i = \max(\alpha_i,\, \psi_i/\varphi_2)$, set $\alpha_i = \max(\alpha_i,\, \psi_i/\varphi_1)$, and go to Step(2).

Step (4): Determine the new estimate α_i by **cubic interpolation**. Set $\alpha_i = \max(\alpha_i,\, \rho_i + \varphi_1(\psi_i - \rho_i))$, set $\alpha_i = \max(\alpha_i,\, \rho_i + \varphi_2(\psi_i - \rho_i))$.

Step (5): If the conditions (5a) and (5c)) are satisfied, with f_{k+1} and g_{k+1} replaced by $f(x_k + \alpha_i s_k)$ and $g(x_k + \alpha_i s_k)$ respectively, then set $\alpha_k = \alpha_i$ and terminate the computation. If both (5a) nd $s_k^T g(x_k + \alpha_i s_k) < 0$, hold then set $\rho_i = \alpha_i$, else set $\psi_i = \alpha_i$; go to **Step(4).**

3.5. Outline of the New Scaled Three-Term CG-Algorithm:

Step 1. Given $x_0 \in R^n$. Let $0 < \delta < \sigma < 1$, $t \ge 0$ and $d_0 = -g_0$. Set $k := 0$.

Step 2. If $\|g_k\| \le 10^{-6}$, then stop.

Step 3. Compute d_k using:

$$d_k = \gamma_k \begin{cases} -g_0, & if \;\; k = 0, \\ -g_k + \beta_k d_{k-1} - (\frac{g_k^T d_{k-1}}{d_{k-1}^T y_{k-1}})\left(y_{k-1} - (2\frac{\|y_{k-1}\|^2}{s_{k-1}^T y_{k-1}})s_{k-1} \right), & if \;\; k \ge 1, \end{cases} \tag{21}$$

γ_k is defined by (20) and β_k is defined in (13b)

Step 4. Find the step-length α_k satisfying cubic line search Algorithm (3.4) and conditions (11) - (12) and set $x_k = x_{k-1} + \alpha_k d_k$.

Step 5. Do a restart step. If ($\beta_k^{PR} < 0$ or $\beta_k^{PR} \le \eta \beta_k^{FR}$ or $\lambda \|g_k\|^2 \le \omega$) then set

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

4. CONVERGENCE RESULTS:

It is well known that any CG-method with perfect line search (with (5c) where $\sigma = 0$) finds the minimum of a quadratic function after at most n steps. This property implies that any convergent CG-method with asymptotically perfect line search and with periodic restart is n-step quadratically convergent [3]. This result is very useful because asymptotically perfect line search can be easily realized by both quadratic and cubic interpolations. The global convergence of CG methods can be assured by suitable restart rules. The simplest such rule is the so-called angle test which consists in setting $\beta_k = 0$ in (3) whenever:

$$\cos(\frac{d_k^T g_k}{\|d_k\|\|g_k\|}) < \delta_0 \tag{22}$$

where δ_0 is a prescribed constant (usually $\delta_0 = 10^{-3}$). A more complicated angle test is proposed in [25]. If the line search is asymptotically perfect, the global convergence of CG-methods can be assured by periodic restarts. The first global convergence result which does not depend on restarts has been obtained by Zoutendijk [34] and Powell [22], who proved that the FR method with perfect line search is globally convergent in the sense that

$$\liminf \|g_k\| = 0 \tag{23}$$

where $\liminf \|g_k\|$ is taken over the iterative process (1). Later, Al-Baali [1] generalized this result to include the FR method without perfect line search. He has shown that (22) holds for the FR method whenever $\sigma < 1/2$ in (5). Recently great effort was devoted to generalizing this result to other CG methods. Touati-Ahmed and Storey [28] have shown that the iterative process (2) and (3) with a line search satisfying (5) is globally convergent if:

$$0 \le \beta_k^{PR} \le \eta \beta_k^{FR} \tag{24a}$$

For $1 < \eta < 1/(2\sigma)$. In this case, we have proposed also the following criterion to ensure the global convergence property, namely:

$$\lambda \|g_k\|^2 \leq \omega \tag{24b}$$

hold in every iteration, where $0 < \lambda$ and $\omega > 0$ are suitable constants.

The proof given in [28] guarantee that, for $\sigma\eta < 1$, the following inequality:

$$g_{k-1}^T d_{k-1} \leq -\frac{1}{1-\sigma\eta} \|g_{k-1}\|^2 \tag{25}$$

is satisfied at every iteration. Therefore the CG-method is a descent one if (24) holds. The most general result has been obtained by Gilbert and Nocedal [10], who have shown that the both PR and HS methods are globally convergent if they generate positive values of β_k and if (25) holds. This result is very important because it allows us to develop a great number of useful restart procedures for CG-methods. The reader may see [19] for more details of some of above theoretical results.

4.1. Lemma: Consider the CG-method in the form (2) and (3), and let the step-length α_k be obtained by the line search Algorithm (3.4) with conditions (11) and (12). Suppose that Assumptions 2.1-2.2 hold. Then one has:

$$\sum_{k=0}^{\infty} \alpha_k^2 \|d_k\|^2 < 0. \tag{26}$$

Proof: Since α_k is obtained by the line search Algorithm (3.4) with conditions (11)-(12). Then, from (8) and (11) we have

$$f_k - f_{k-1} \leq -\delta \alpha_k^2 \|d_k\|^2 \leq 0. \tag{27}$$

Hence, $\{f_k\}$ is a decreasing sequence and the sequence $\{x_k\}$ is contained in F. Hence, Assumptions 2.1-2.2 imply that there exists a constant f^* such that:

$$\lim_{k\to\infty} f_k = f^*. \tag{28}$$

From (28), we have:

$$\sum_{k=0}^{\infty} (f_{k-1} - f_k) < +\infty. \tag{29}$$

This together with (27) implies that (26) holds.

4.2. Lemma: For the new proposed algorithm, defined in (21), if there exists a constant $\varepsilon > 0$ such that:

$$\|g_k\| \geq \varepsilon, \quad \forall k \geq 0, \tag{30}$$

then there exists a constant $M > 0$ such that

$$\|d_k\| \leq M, \quad \forall k \geq 0. \tag{31}$$

Proof: The proof is same as in [32] except that we have to prove:

$$t = 2\frac{\|y_{k-1}\|^2}{s_{k-1}^T y_{k-1}} = \frac{2}{\alpha_{k-1}} \frac{\|y_{k-1}\|^2}{d_{k-1}^T y_{k-1}} \geq 0 \tag{32a}$$

and

$$\gamma_k = \frac{y_{k-1}^T s_{k-1}}{y_{k-1}^T y_{k-1}} \geq 0, \quad \text{since } y_{k-1}^T s_{k-1} > 0, \quad \text{(B is positive definite)} \tag{32b}$$

From the line search conditions (11)-(12) and (8), we have

$$y_{k-1}^T d_{k-1} = g_k^T d_{k-1} - g_{k-1}^T d_{k-1} \geq -(1-\sigma) g_{k-1}^T d_{k-1} = (1-\sigma)\|g_{k-1}\|^2. \tag{33}$$

since $0 < \sigma < 1$ implies $(1-\sigma)$ is positive. Since α_{k-1} is also positive step-size obtained by a line search producer, hence the parameter t defined in (13c) is positive. This will complete the proof of lemma 4.2. Using the preceding lemmas, we are now ready to give the final convergence results.

4.3. Theorem: Suppose that Assumptions 2.1-2.2 hold. Let $\{x_k\}$ be a sequence of points generated by the new proposed algorithm defined by (21). Then one has

$$\lim_{k\to\infty} \inf \|g_k\| = 0. \tag{34}$$

Proof: We proceed by contradiction. Assume that the conclusion is not true, then there exists a positive constant ε such that

$$\|g_k\| \geq \varepsilon, \quad \forall k \geq 0. \tag{35}$$

If $\liminf_{k\to\infty} \alpha_k > 0$, we have from (26) that $\liminf_{k\to\infty} \|g_k\| = 0$. This contradicts assumption (35). Suppose that $\liminf_{k\to\infty} \alpha_k > 0$. Using Assumptions 2.1-2.2 and from condition (12a), we obtain:

$$-(1-\sigma) g_k^T d_k \leq (g_{k+1} - g_k)^T d_k \leq L\alpha_k \|d_k\|^2. \tag{36}$$

Combining with (8) yields:

$$(1-\sigma)\|g_k\|^2 \leq L\alpha_k \|d_k\|^2. \tag{37}$$

The above inequality and Lemma 4.2 imply $\liminf_{k\to\infty} \|g_k\| = 0$, which contradicts (35). This completes the proof.

5. NUMERICAL RESULTS.

The main work of this section is to report the performance of the new proposed algorithm (say NEW) on a set of test problems. The codes were written in Fortran77 and in double precision arithmetic. All the tests were performed on a PC. Our experiments were performed on a set of (35) nonlinear unconstrained problems that have second derivatives available. These test problems are contributed in CUTE [4] and their details are given in the Appendix. For each test function we have considered 10 numerical experiments with number of variables *n* = 100, 200, . . . , 1000 and we have reported the total amount of each test problem. In order to assess the reliability of our new proposed method, we have tested it against the proposed Al-Bayati and Altae three-term CG-method [2] using the same test problems. All these methods terminate when the following stopping criterion is met:

$$\|g_k\| \leq 10^{-6}. \tag{38}$$

We also force these routines stopped if the iterations exceed 1000 or the number of function evaluations reach 2000 without achieving convergence. We use Algorithm (3.4) as the line search routine satisfying (11) and (12). Tables 5.1 compares some numerical results for NEW method against Al-Bayati and Al-tae [2] three-term CG-method; this table indicates for (n) as a dimension of the problem; (NOI), number of iterations; (NOFG), number of function and gradient evaluations; (TIME), the total time required to complete the evaluation process for each test problem. In Table 5.2 we have compared the percentage performance of the NEW method against Al-Bayati and Altae three-term CG-method [2] taking over all the tools as 100%.

Table 5.1. Numerical Results for NEW-Algorithm against Algorithm (2.5):
For the total of (35) test problems

Algorithm (2.5)					NEW Algorithm (3.5)				
n	NOI	NOFG	TIME	Gmin	n	NOI	NOFG	TIME	Gmin
1 TOTAL (seconds)	438	805		0.43	1 TOTAL (seconds)	430	767		0.40
2 TOTAL (seconds)	358	4643		0.30	2 TOTAL (seconds)	330	4259		0.22
3 TOTAL (seconds)	40	90		0.03	3 TOTAL (seconds)	40	90		0.03
4 TOTAL (seconds)	4120	6102		2.33	4 TOTAL (seconds)	2809	4203		2.49
5 TOTAL (seconds)	511	5846		0.42	5 TOTAL (seconds)	504	5801		0.40
6 TOTAL (seconds)	215	436		0.07	6 TOTAL (seconds)	232	444		0.06
7 TOTAL (seconds)	335	6178		2.96	7 TOTAL (seconds)	301	3590		1.44
8 TOTAL (seconds)	143	293		0.04	8 TOTAL (seconds)	198	361		0.05
9 TOTAL (seconds)	40	90		0.06	9 TOTAL (seconds)	40	90		0.06
10 TOTAL (seconds)	245	444		0.03	10 TOTAL (seconds)	247	440		0.04
11 TOTAL (seconds)	211	2104		0.91	11 TOTAL (seconds)	213	2170		0.98
12 TOTAL (seconds)	615	967		0.20	12 TOTAL (seconds)	537	898		0.18
13 TOTAL (seconds)	20	50		0.02	13 TOTAL (seconds)	20	50		0.02
14 TOTAL (seconds)	109	223		0.04	14 TOTAL (seconds)	102	208		0.03
15 TOTAL (seconds)	111	232		0.06	15 TOTAL (seconds)	111	232		0.06
16 TOTAL (seconds)	149	272		0.07	16 TOTAL (seconds)	148	270		0.07
17 TOTAL (seconds)	473	910		0.10	17 TOTAL (seconds)	503	983		0.10
18 TOTAL (seconds)	661	8083		0.75	18 TOTAL (seconds)	580	6812		0.58
19 TOTAL	149	394		0.03	19 TOTAL	149	394		0.03

(seconds)	(seconds)
20 TOTAL 374 744 0.08 (seconds)	20 TOTAL 395 774 0.08 (seconds)
21 TOTAL 40 90 0.03 (seconds)	21 TOTAL 40 90 0.03 (seconds)
22 TOTAL 828 14980 1.02 (seconds)	22 TOTAL 776 13203 0.98 (seconds)
23 TOTAL 246 458 0.15 (seconds)	23 TOTAL 238 440 0.15 (seconds)
24 TOTAL 201 412 0.05 (seconds)	24 TOTAL 201 412 0.05 (seconds)
25 TOTAL 272 488 0.06 (seconds)	25 TOTAL 233 421 0.06 (seconds)
26 TOTAL 200 432 0.05 (seconds)	26 TOTAL 200 432 0.05 (seconds)
27 TOTAL 39 108 0.03 (seconds)	27 TOTAL 39 108 0.03 (seconds)
28 TOTAL 61 824 0.30 (seconds)	28 TOTAL 61 824 0.30 (seconds)
29 TOTAL 30 70 0.03 (seconds)	29 TOTAL 30 70 0.03 (seconds)
30 TOTAL 211 2104 0.91 (seconds)	30 TOTAL 153 1370 0.84 (seconds)
31 TOTAL 1404 1794 0.81 (seconds)	31 TOTAL 489 900 0.08 (seconds)
32 TOTAL 10 30 0.01 (seconds)	32 TOTAL 10 30 0.01 (seconds)
33 TOTAL 355 675 0.09 (seconds)	33 TOTAL 353 663 0.07 (seconds)
34 TOTAL 90 110 0.02 (seconds)	34 TOTAL 90 110 0.02 (seconds)
35 TOTAL 195 437 0.05 (seconds)	35 TOTAL 288 478 0.06 (seconds)
Total Of 35 14132 61954 12.54 Test Fun.	Total Of 35 11090 52387 10.08 Test Fun.

Table 5.2. Percentage performance of NEW-Algorithm against Algorithm (2.5)

Tools	**Algorithm (2.5)**	**NEW (Algorithm 3.5)**
NOI	100%	78.5%
NOFG	100%	84.5%
TIME	100%	80.3%

It is clear from Table (5.2) that taking, over all, the Tools as a 100% for the Al-Bayati and Altae three-term CG-method, namely algorithm (2.5), the NEW-Algorithm has an improvement, in about (21.5%) NOI; (16.5%) NOFG and (19.7%) TIME.

CONCLUSIONS:

Taking everything into consideration the new proposed scaled three-term CG-method have been obtained very significant development as we have expected, we think that, for all the specific problems, the enhancement of the new proposed method is very robust. However, we know that CG-methods are sensitive to the order of interpolation; therefore, we have recommend a modified perfect cubic interpolation over the standard quadratic one in our implementations. Hence, we believe that the new method is a valid approach for the problems and has its own potential. Also, the effectiveness of this new proposed method depends on the robustness set of selected scaling criteria and several selected sets of restating techniques used in this research besides the selected set of test functions.

ACKNOWLEDGEMENTS

We are indebted to B. Albaro for kindly sharing with us her findings in this subject and completing the layout of this manuscript.

APPENDIX

The details of the test functions, used in this paper, can be found in [4]. The numbers (1-35) in Table 5.1 indicate to:

1- Extended Trigonometric Function.
2- Extended Penalty Function.
3- Raydan 2 Function.
4- Diagonal2 Function.
5- Generalized Tridiagonal-1 Function.
6- Extended Tridiagonal-1 Function.
7- Extended 3-Exponential Terms Function.
8- Diagonal4 Function.
9- Diagonal5 Function.
10- Extended Himmelblau Function.
11- Extended PSC1 Function.
12- Extended Block Diagonal BD1 Function.
13- Extended EP1 Function.
14- DIXMAANA CUTE- Function.
15- DIXMAANB CUTE- Function.
16- DIXMAANC CUTE- Function.
17- Broyden Tri-diagonal Function.
18- EDENSCH CUTE- Function.
19- VARDIM CUTE- Function.
20- LIARWHD CUTE- Function.
21- DIAGONAL 6 Function.
22- ENGVAL1 CUTE- Function.
23- DENSCHNA CUTE- Function.
24- DENSCHNB CUTE- Function.
25- DENSCHNF CUTE- Function.
26- Generalized Quartic GQ1 function.
27- Diagonal 7 Function.
28- Diagonal 8 Function.
29- Full Hessian Function.
30- SINCOS Function.
31- Generalized quartic GQ2 function.
32- ARGLINB CUTE-Function.
33- FLETCHCR CUTE-Function.
34- HIMMELBG CUTE-Function.
35- HIMMELBH CUTE-Function.

REFERENCES

1. M. Al-Baali, Descent property and global convergence of the Fletcher-Reeves method with inexact line search, IMA Journal of Numerical Analysis, 5(1) (1985), pp. 121–124.
2. A. Y. Al-Bayati and W. H. Altae, A New Three-Term Non-Linear Conjugate Gradient Method For Unconstrained Optimization, Canadian Journal on Science & Engineering Mathematics, SEM-1009-011, AM Publishers Corporation, Canada, 1(5) (2010), pp. 108-124.
3. P. Baptist and J. Stoer, On the relation between quadratic termination and convergence properties of minimization algorithms, Part 2. Applications. Numer. Math., 28 (1977), pp. 367-391.
4. I. Bongartz, A. R. Conn, N. Gould, and P. L. Toint, CUTE: constrained and unconstrained testing environment, ACM Transactions on Mathematical Software, 21(1) (1995), pp. 123–160.
5. Y. H. Dai and Y. Yuan, Convergence properties of the Fletcher-Reeves method, IMA Journal of Numerical Analysis, 16(2) (1996), pp. 155–164.
6. Y. H. Dai and Y. Yuan, A nonlinear conjugate gradient method with a strong global convergence property, SIAM Journal on Optimization, 10(1) (2000), pp. 177–182.
7. Y. H. Dai, and Y. Yuan, Nonlinear Conjugate Gradient Methods, Shang-Hai Science and Technology, Beijing, 2000.
8. R. S. Dembo and T. Steihaug, Truncated-Newton algorithms for large-scale unconstrained minimization, Math. Programming, 26 (1983), pp. 190-212.

9. R. Fletcher and C. M. Reeves, Function minimization by conjugate gradients, Computer J., 7 (1964), pp. 149-154.
10. J. C. Gilbert and J. Nocedal, Global convergence properties of conjugate gradient methods for optimization, Report No. 1268, Institut National de Recherche en Informatique et en Automatique, 1990.
11. J. C. Gilbert and J. Nocedal, Global convergence properties of conjugate gradient methods for optimization, SIAM Journal on Optimization, 2(1) (1992), pp. 21–42.
12. A. Griewank and P. L. Toint, Partitioned variable metric updates for large structured optimization problems, Numer. Math., 39 (1982), pp. 119-137.
13. L. Grippo and S. Lucidi, A globally convergent version of the Polak-Ribi`ere conjugate gradient method, Mathematical Programming, 78(3) (1997), pp. 375–391.
14. L. Grippo and S. Lucidi, Convergence conditions, line search algorithms and trust region implementations for the Polak Ribiere conjugate gradient method, Optimization Methods & Software, 20(1) (2005), pp. 71–98.
15. W. W. Hager, A derivative-based bracketing scheme for univariate minimization and the conjugate gradient method, Comput. Math. Appl, 18 (1989), pp. 779–795.
16. W. W. Hager and H. Zhang, A survey of nonlinear conjugate gradient methods, Pacific Journal of Optimization, 2 (2006), pp. 35–58.
17. M. R. Hestenes and E. Stiefel, Methods of conjugate gradient for solving linear systems, Journal of Research of the National Bureau of Standards, 49 (1952), pp. 409–436.
18. L. Luk¹an, Computational experience with improved variable metric methods for unconstrained minimization, Kybernetika, 26 (1990), pp. 415-431.
19. L. Luk¹an, Computational Experience with Improved Conjugate Gradient Methods for Unconstrained Minimization, Technical Report, V-488, 1991.
20. J. Nocedal, Updating quasi-Newton Matrices with limited storage, Math. Comput., 35 (1980), pp. 773-782.
21. E. Polak and G. Ribiere, Note sur la convergence de directions conjug´ees, Revue Francaised ’Informatique et de Recherche Operationnelle, 16 (1969), pp. 35–43.
22. M. J. D. Powell, Non-convex minimization calculations and the conjugate gradient method, Report No. DAMTP 1983/NA14, University of Cambridge, 1983.
23. M. J. D. Powell, Convergence properties of algorithms for nonlinear optimization, Report No. DAMPT 1985/NA1, University of Cambridge, 1985.
24. D. F. Shanno, Conditioning of quasi-Newton methods for function minimization, Math. Comput. 24 (1970), pp. 647-656.
25. D. F. Shanno, Globally convergent conjugate gradient algorithms, Math. Programming, 33 (1985), pp. 61-67.
26. J. Sun and J. Zhang, Global convergence of conjugate gradient methods without line search, Annals of Operations Research, 103 (2001), pp. 161–173.
27. P. L. Toint, On sparse and symmetric matrix updating subject to a linear equation, Math. of Comp., 31 (1997), pp. 954- 961.
28. D. Touati-Ahmed and C. Storey, Efficient hybrid conjugate gradient techniques, J. Optim. Theory Appl., 64 (1990), pp. 379-397.
29. Z. Wei, G. Y. Li, and L. Qi, Global convergence of the Polak-Ribi`ere-Polyak conjugate gradient method with an Armijo-type inexact line search for non-convex unconstrained optimization problems, Mathematics of Computation, 77(264) (2008), pp. 2173–2193.
30. P. Wolfe, Convergence conditions for ascent methods, SIAM Rev, 11 (1969), pp. 26–235.
31. P. Wolfe, Convergence conditions for ascent methods II: Some corrections, SIAM Rev, 13 (1995), pp.185–188.
32. J. Zhang, Y. Xiao and Z. Wei, Nonlinear conjugate gradient methods with sufficient descent condition for large scale unconstrained optimization, Mathematical Problems in Engineering, 16 pages, 2009. DOI: 1155/2009/243290.
33. L. Zhang, W. Zhou, and D. H. Li, Global convergence of a modified Fletcher-Reeves conjugate gradient method with Armijo-type line search, Numerische Mathematik, 104(4) (2006), pp. 561–572.
34. G. Zoutendijk, Nonlinear programming, computational methods, In: "Integer and Nonlinear Programming" (J. Abadie ed.), North-Holland, Amsterdam, (1970), pp. 93-121.

Source of support: Nil, Conflict of interest: None Declared