*International Journal of*
*Mathematical Modelling & Computations*
Vol. 10, No. 04, Fall 2020, 295- 309

# An Optimum Line Search for Unconstrained Non-Polynomial Test Functions Using Nonlinear Conjugate Gradient Methods

A. A. Ishaq[a], T. Latunde[b,*] and F. M. Jimoh[c]

[a,c]*Department of Physical Sciences, Alhikmah University, Ilorin, Nigeria,*
[b]*Department of Mathematics, Federal University Oye-Ekiti, Oye-Ekiti, Nigeria.*

**Abstract.** The nonlinear conjugate gradient method solves issues of the frame:

$$\text{minimize} f(\underline{x}), \underline{x} \in \Re$$

employing an iterative plot, $\underline{x}^{(k+1)} = \underline{x}^{(k)} + \alpha_k \underline{d}^{(k)}$, where $f$ is a non-polynomial function. We utilized two variants of the optimum line search namely, direct and indirect methods, to compute the step-length in this paper. Both line searches yielded great outcome when employed to a few unconstrained non-polynomial test functions.

Received: 07 July 2020, Revised: 12 October 2020, Accepted: 23 October 2020.

**Keywords:** Nonlinear conjugate gradient method; Step-length; Line search; Unconstrained optimization problem; Large scale problems.

**AMS Subject Classification**: 90C30.

**Index to information contained in this paper**



## 1. Introduction

We intend to observe improvements in large-scale optimization recently. And like to mention in any case that small-scale optimization is still a dynamic region of investigation, and progresses in this domain of knowledge frequently translate into

*Corresponding author. Email: tolulope.latunde@fuoye.edu.ng

a new line of solutions for large-scale problems.
The problem beneath thought is the following unconstrained optimization issue:

$$\min f(x), x \in \Re^n \tag{1}$$

where $f$ is a smooth function of $n$ variables, $\Re^n$ is an n-dimensional Euclidean vector space. When $n$ is large, say $n \geqslant 5000$, then (1) is called a large-scale unconstrained optimization problem. These problems do emerge in numerous sectors namely mining, economics, engineering, telecommunications, business, manufacturing, energy, military planning, medicine, science and so on.

## 2. Methods of solution

Several strategies are accessible for tackling unconstrained optimization problem (1) [26]. These strategies are gathered into two diverse groups as direct search strategies and gradient-based strategies.

### 2.1 *Direct search strategies*

These are techniques for tackling optimization issues which any information about the gradient of $f$ is not required [21]. Unlike the conventional optimization methods which utilize information about the gradient or higher derivatives to obtain the optimum, these strategies solve an optimization problem by finding a set of points close to the present point, then seeking for one where the value of $f$ is lesser than the value at the present point. The methods solve test function that is not differentiable, or not continuous and they are most suitable for small scale problems. Examples of direct search strategies are simplex search method, Hooke and Jeeves method, Rosenbroock method and so on.

### 2.2 *Gradient-based strategies*

These types of methods which is also known as descent methods, seek the calculation of first-order and possibly higher order partial derivatives of $f$ with respect to each of its variables $x^{(1)}, x^{(2)}, \cdots x^{(n)}$ and it is written as $\nabla f(x)$. Thus,

$$\nabla f(x) = (\frac{\partial f(x)}{\partial x_1}, \frac{\partial f(x)}{\partial x_2}, \ldots, \frac{\partial f(x)}{\partial x_n})^T, \tag{2}$$

$\nabla f(x)$ does possess an exceptionally vital property in it which portrays steepest ascent direction [20]. Regarding this relevant property, the negative of (2) is referred to as the steepest descent direction. Therefore, any strategy which makes use of (2) can be anticipated to examine the least point quicker than the one which does not utilize it. Examples of gradient-based strategies that solve (1) are the Newton method, quasi-Newton method, variable metric method, conjugate gradient methods and so on ([17], [1]). Hence, all the gradient methods that shall be discussed require the gradient vector in finding the search direction.

#### *2.2.1 Newton method*

In optimization, this approach, named after Isaac Newton and Joseph Raphson, is a strategy applied to find the roots of the derivative i.e., $\nabla f(x) = 0$, which is also referred as the stationary points of $f$ [25]. The strategy is well known and broadly utilized due to its capability of giving the most accurate answers when solving small-scale optimization problems [18]. The method converges exceptionally quickly but the convergence can not be ensured because it needs expensive computing memory facility and may not continuously provide slightest computing time depending on the traits of the problem.

#### *2.2.2 Quasi-Newton method*

These methods are strategies utilized to obtain either zeroes or local extrema of the objective functions, this serves as an alternative strategy to Newton's method [27]. The strategy can be employed if the Hessian is unavailable or is as well costly to compute at each iteration. The major drawback of Newton's method even after modification to ensure convergence is that of the calculation of the second derivative of $f$ at every iteration. Thus, quasi-Newton approaches were introduced to define efficient and very effective methods which do not involve the calculation of the hessian during the computations [22].

#### *2.2.3 Variable metric method*

This strategy was first introduced in 1959 and later extended in 1963 ([7], [9]). Variable metric methods are commonly utilized in connection with unconstrained optimization since they have good theoretical and convergence properties. They are regarded as the best general unconstrained optimization technique and the method makes use of the derivative that is currently available.

#### *2.2.4 Conjugate gradient method (CGM)*

The method CG has been an area of active research since 1952 [12]. This approach is a capable one for solving optimization problems due to the conciseness of its algebraic expression, simplicity of its analysis and ease of its implementation. And due to its very moo memory necessity, quick convergence and so on, it remains exceptionally prevalent for researchers who are curious in tackling large-scale optimization issues ([13], [14], [23]).

CGMs are originally introduced for the solution of a strictly convex quadratic function in the form

$$f(x) = f_0 + x^T a + \frac{1}{2} x^T A x \tag{3}$$

where $f_0 \in \Re$, $x, a \in \Re^n$ and $A \in \Re^{n*n}$ with the aim of accelerating its convergence compared to the previously discussed methods.

### 2.3 *Nonlinear CGMs*

When the method CG is employed for solving non-quadratic functions, it is called nonlinear CGM. For quadratic objective function defined in (3), the Hessian, A, is constant. In any case, for a common nonlinear function, the Hessian is a matrix which needs to be assessed at every iteration. This could be computationally costly. Hence, efficient execution of the CG algorithm that disposes the Hessian evaluation at each step is desirable [15].

The nonlinear CGM is a computational procedure that solves the unconstrained optimization problem in (1) efficiently. It uses an iterative scheme of the form:

$$x^{(k+1)} = x^{(k)} + \alpha_k d^{(k)} \tag{4}$$

where, $x^{(k+1)}$ is the new iterate point to be obtained, $x^{(k)}$ is the present iterate point, $\alpha_k > 0$ is the step-length which can be determined by various step-length rules and $d^{(k)}$, the search directions that are calculated by the rule

$$d^{(k)} = \begin{cases} -g^{(k)} + \beta_{(k-1)} d^{(k-1)}; k \geqslant 1 \\ \qquad\qquad -g^{(k)}; \quad k = 0 \end{cases} \tag{5}$$

where $g^{(k)} = \nabla f(x^{(k)})$ and $\beta_{(k)}$ is a scalar referred as the CG update parameter ([4] and [3]). Some common formulae for $\beta_k$ are given as Fletcher-Reeves (FR), Polak-Ribiere-Polyak (PRP), Hestensee-Stifel (HS), Conjugate-Descent (CD), Dai-Yuan (DY), Liu-Storey (LS), Bamgbola-Ali-Nwaeze (BAN), Hager-Zhang (HZ), respectively defined by,

$$\beta_k^{FR} = \frac{\left\|g^{(k+1)}\right\|^2}{\left\|g^{(k)}\right\|^2}, [10] \tag{6}$$

$$\beta_k^{PRP} = \frac{g^{(k+1)T} y^{(k)}}{\left\|g^{(k)}\right\|^2}, [19] \tag{7}$$

$$\beta_k^{HS} = \frac{g^{(k+1)T} y^{(k)}}{g^{(k)T} g^{(k)}}, [12] \tag{8}$$

$$\beta_k^{CD} = -\frac{\left\|g^{(k+1)}\right\|^2}{g^{(k)T} d^{(k)}}, [8] \tag{9}$$

$$\beta_k^{DY} = \frac{\left\|g^{(k+1)}\right\|^2}{y^{(k)T} d^{(k)}}, [6] \tag{10}$$

$$\beta_k^{LS} = -\frac{g^{(k+1)T} y^{(k)}}{g^{(k)T} d^{(k)}}, [16] \tag{11}$$

$$\beta_k^{BAN} = \frac{g^{(k+1)T} y^{(k)}}{g^{(k)T} y^{(k)}}, [5] \tag{12}$$

$$\beta_k^{HZ} = \left( y^{(k)} - 2d^{(k)} \frac{\left\|y^{(k)}\right\|^2}{y^{(k)T} d^{(k)}} \right)^{(T)} \frac{g^{(k+1)T} d^{(k)}}{y^{(k)T} d^{(k)}}, [11]. \tag{13}$$

If $f$ is given as in (3) and the line search is optimum, then the approach (4) and (5) is referred as the linear CGM [24]. For the linear CGM, parameters (6)-(13) are identical [1]. However, for the nonlinear CGM, parameters (6)-(13) lead to diverse performance in practice which results in distinct CGMs ([28], [29]).

## 3. Optimum line search

The purpose of every line search rule is to get a positive $\alpha_k$ alongst the $d^{(k)}$ on the objective of guarantee an improving speed of convergence. Here, we consider the approaches involve in calculating $\alpha_k$ using an optimum line-search. Then to accomplish this, we first set $\alpha_k = \alpha^*$, for,

$$\alpha^* = \arg\ \min_{\alpha>0} f(x^{(k)} + \alpha d^{(k)}) \tag{14}$$

i.e., $\alpha^*$ is the value of $\alpha_k$ which minimizes $f$ along $d^{(k)}$ [17]. Therefore, $\alpha^*$ in (14) can be determined by solving the following equation,

$$\frac{d}{d\alpha} f(x^{(k)} + \alpha d^{(k)}) = 0 \tag{15}$$

The strategies employed in (15) results to an exact or optimum value for $\alpha_k$ and this is called an exact or optimum line search.

This work examines the numerical processes required in executing the optimum line search for minimizing non-polynomial optimization problems only, using two different approaches. The two approaches and their illustrations are discussed next.

### 3.1 *Solution by conversion*

This is also known as indirect approach for solving non-polynomial functions. For a test function, we expand the function in Taylor's series, then truncate the series after a desired number of terms. Below is an example for the illustration.

***Raydan 2 function***

$$f(\underline{x}) = \sum_{i=1}^{n} [\exp(x_i) - x_i], \;\; \underline{x}^{(0)} = [1, 1, ..., 1].$$

Solution

$$f(\underline{x}) = \sum_{i=1}^{n} [1 + x_i + \frac{x_i^2}{2!} + \frac{x_i^3}{3!} + \frac{x_i^4}{4!} + ... - x_i]$$

Truncated form for the quatic: $f(\underline{x}) = \sum_{i=1}^{n} [1 + \frac{x_i^2}{2!} + \frac{x_i^3}{3!} + \frac{x_i^4}{4!}]$
Then the gradient is $\underline{g}(x_i)|_{i=1}^{n} = \frac{\partial}{\partial x_i} f(\underline{x}) = x_i + \frac{x_i^2}{2} + \frac{x_i^3}{6}$
By using the technique of optimum line search discussed earlier to calculate $\alpha_k$, i.e.,

$$f(\underline{x}) = f(\underline{x}_i + \;\alpha \underline{d}) = f(\alpha)$$

$$\Rightarrow f(x_i + \alpha d_i) = \sum_{i=1}^{n} \left[1 + \frac{(x_i+\alpha d_i)^2}{2} + \frac{(x_i+\alpha d_i)^3}{3} + \frac{(x_i+\alpha d_i)^4}{24}\right]$$

$$= \sum_{i=1}^{n} \left[1 + \frac{x_i^2}{2} + \alpha x_i d_i + \alpha^2 \frac{d_i^2}{2} + \frac{x_i^3}{3} + \alpha x_i^2 d_i \alpha^2 x_i^2 d_i^2 + \frac{\alpha^3 d_i^3}{3} + \frac{x_i^4}{24}\right.$$
$$\left. + \frac{\alpha x_i^3 d_i}{6} + \frac{\alpha^2 x_i^2 d_i^2}{4} + \frac{\alpha^3 x_i d_i^3}{6} + \frac{\alpha^4 d_i^4}{24}\right]$$

$$= \sum_{i=1}^{n} \left[1 + \frac{x_i^2}{2} + \frac{x_i^3}{3} + \frac{x_i^4}{24}\right] + \alpha \sum_{i=1}^{n} \left[x_i d_i + x_i^2 d_i + \frac{x_i^3 d_i}{6}\right]$$
$$+ \alpha^2 \sum_{i=1}^{n} \left[\frac{d_i^2}{2} + x_i d_i^2 + \frac{x_i^2 d_i^2}{4}\right] + \frac{\alpha^3}{3} \sum_{i=1}^{n} \left[d_i^3 + \frac{x_i d_i^3}{2}\right] + \frac{\alpha^4}{24} \sum_{i=1}^{n} d_i^4$$

Since $\frac{\partial f}{\partial \alpha}(x_i + \alpha d_i) = 0$,

$$\sum_{i=1}^{n}\left(x_i d_i + x_i^2 d_i + \frac{x_i^3 d_i}{6}\right) + 2\alpha \sum_{i=1}^{n}\left(\frac{d_i^2}{2} + x_i d_i^2 + \frac{x_i^2 d_i^2}{4}\right) + \alpha^2 \sum_{i=1}^{n}\left(d_i^3 + \frac{x_i d_i^3}{2}\right) + \frac{\alpha^3}{6}\sum_{i=1}^{n} d_i^4 = 0.$$

Hence, we obtained our result using matlab program by imputing the following values for $f(\underline{x})$, $g_i(\underline{x})$, $f(\alpha)$ and $\underline{x}^{(0)}$ with eight different conjugate gradient methods. And the result we obtained from the example above is shown in Table 1.

Table 1. Numerical solution for indirect method.

| CGMs | n. (Dim) | Indirect Method | | | Exc. Time |
|---|---|---|---|---|---|
| | | Itr | f* | Norm g* | |
| BAN | 5000 | 1 | 5.00e003 | 2.7e-011 | 0.10 |
| | 10000 | 1 | 1.00e.004 | 3.0e-011 | 0.08 |
| FR | 5000 | 1 | 5.00e003 | 2.7e-o11 | 0.05 |
| | 10000 | 1 | 1.00e.004 | 3.0e-011 | 0.09 |
| PR | 5000 | 1 | 5.00e003 | 2.7e-o11 | 0.07 |
| | 10000 | 1 | 1.00e.004 | 3.0e-011 | 0.09 |
| HS | 5000 | 1 | 5.00e003 | 2.7e-o11 | 0.07 |
| | 10000 | 1 | 1.00e.004 | 3.0e-011 | 0.09 |
| CD | 5000 | 1 | 5.00e003 | 2.7e-o11 | 0.07 |
| | 10000 | 1 | 1.00e.004 | 3.0e-011 | 0.09 |
| DY | 5000 | 1 | 5.00e003 | 2.7e-o11 | 0.07 |
| | 10000 | 1 | 1.00e.004 | 3.0e-011 | 0.10 |
| LS | 5000 | 1 | 5.00e003 | 2.7e-o11 | 0.07 |
| | 10000 | 1 | 1.00e.004 | 3.0e-011 | 0.10 |
| HZ | 5000 | 1 | 5.00e003 | 2.7e-o11 | 0.07 |
| | 10000 | 1 | 1.00e.004 | 3.0e-011 | 0.10 |

### 3.2 *Direct method*

This is an approach by applying the CG Algorithm on the given problem directly to obtain the required solutions. We will also use the same example above for the illustration of this method.

*Raydan 2 function*

$$f(x) = \sum_{i=1}^{n} [\exp(x_i) - x_i], \quad x^{(0)} = [1, 1, ..., 1].$$

Solution

$$\underline{g}_i(\underline{x}) = \frac{\partial}{\partial x_i} f(\underline{x}) = \begin{bmatrix} e^{x_1} - 1 \\ e^{x_2} - 1 \\ \vdots \\ e^{x_n} - 1 \end{bmatrix}$$

$$\underline{g}^{(0)} = \begin{bmatrix} e^1 - 1 \\ e^1 - 1 \\ \vdots \\ e^1 - 1 \end{bmatrix}, \quad \underline{d}^{(0)} = -\underline{g}^{(0)} = - \begin{bmatrix} e^1 - 1 \\ e^1 - 1 \\ \vdots \\ e^1 - 1 \end{bmatrix}$$

$$\begin{aligned} \alpha_0 &= \arg\ \min f(\underline{x}^{(0)} + \alpha \underline{d}^{(0)}) \\ &= \arg\ \min f \begin{bmatrix} 1 \\ 1 \\ \vdots \\ 1 \end{bmatrix} - \alpha \begin{bmatrix} e^1 - 1 \\ e^1 - 1 \\ \vdots \\ e^1 - 1 \end{bmatrix} \\ &= \min_\alpha \sum_{i=1}^{n} [e^{1+\alpha-\alpha e^1} - (1 + \alpha - \alpha e^1)] \end{aligned}$$

i.e.,

$$\begin{aligned} &\frac{d}{d\alpha}[n(e^{1+\alpha_0-\alpha_0 e^1}) - (1 + \alpha_0 - \alpha_0 e^1)] = 0 \\ &\Rightarrow n[(1 - e^1)e^{1+\alpha_0-\alpha_0 e^1} - (1 - e^1)] = 0 \\ &\Rightarrow (1 - e^1)[e^{1+\alpha_0-\alpha_0 e^1} - 1] = 0 \\ &\Rightarrow e^{1+\alpha_0-\alpha_0 e^1} - 1 = 0 \\ &\Rightarrow e^{1+\alpha_0-\alpha_0 e^1} = 1 = e^0 \\ &\Rightarrow 1 + \alpha_0 - \alpha_0 e^1 = 0 \end{aligned}$$

i.e., $\alpha_0 = \frac{1}{e^1-1}$ so,

$$\underline{x}^{(0)} = \underline{x}^k + \alpha_k \underline{d}^k$$

$$\underline{x}^1 = \begin{bmatrix} 1 \\ 1 \\ \vdots \\ 1 \end{bmatrix} - \frac{1}{(e-1)} \begin{bmatrix} (e-1) \\ (e-1) \\ \vdots \\ (e-1) \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix}$$

$$\underline{g}^{(1)} = \underline{g}(x^1) = \begin{bmatrix} e^{(0)} - 1 \\ e^{(0)} - 1 \\ \vdots \\ e^{(0)} - 1 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix}$$

Thus, $\underline{x}^* = \underline{x}^{(1)}$. So, substituting $x^*$ in the function $f(\underline{x})$ then, $f^* = n$. Hence the result for the example are tabulated below:

Table 2. Numerical solution for direct method.

| CE | n. (Dim) | Direct Method | | |
|---|---|---|---|---|
| | | Itr | $f^*$ | $\parallel g^* \parallel$ |
| 1 | 5000 | 1 | $5.0 * 10^3$ | $\underline{0}$ |
| | 10000 | 1 | $1.0 * 10^4$ | $\underline{0}$ |

## 4. Numerical experiments

### 4.1 *Nonlinear CGMs algorithm for optimum line search*

**Step 1:** Pick the starting point, $x^{(0)} \in \mathbb{R}^n$, $\epsilon \geq 0$ (a small number called tolerance) and set $d^{(0)} = -g^{(0)} = -\nabla f(x^{(0)}), k = 0$.
**Step 2:** Terminate process if $||g^{(0)}|| \leqslant \epsilon$, else, go to Step 3.
**Step 3:** Calculate $\alpha_k = \arg\ \min f(\underline{x}^{(k)} + \alpha \underline{d}^{(k)})$, $\alpha > 0$.
**Step 4:** Set $x^{(k+1)} = x^{(k)} + \alpha_k d^{(k)}$; if $||g^{(k+1)}|| \leqslant \epsilon$, stop, else, move to the next step.
**Step 5:** Find $\beta_k$ for the chosen CGM and $\underline{d}^{(k)}$ using (5).
**Step 6:** Set $k = k + 1$, and move back to Step 3.

### 4.2 *Computational details*

Our aim is to perform the experiments using nonlinear CGMs to minimize non-polynomial unconstrained optimization problems. To achieve this, ten test functions extracted from [2] were utilized as numerical examples. The test functions are mainly non-polynomial objective functions.
The nonlinear CGMs Algorithm 4.1 for optimum line search was implemented using indirect method and direct method in Subsections 3.1 and 3.2 respectively. It suffices to say here that the dimensionality of the test functions were generally taken to be very large (5000 and 10000). Also, we have assumed a tolerance of $10^{-6}$ for the norm of $g^*$ (the gradient at the optimum value of $f$).

### 4.3 *Computational examples*

#### *4.3.1 Diagonal 3*

$$f(\underline{x}) = \sum_{i=1}^{n} [\exp(x_i) - \sin(x_i)], \quad \underline{x}^{(0)} = [1, 1, ...1]^T$$

#### 4.3.2 Full Hessian FH3

$$f(\underline{x}) = \sum_{i=1}^{n} x^2 + \sum_{i=1}^{n} [x_i \text{exp}(x_i) - 2x_i - x_i^2], \quad \underline{x}^{(0)} = [1, 1, ...1]^T$$

#### 4.3.3 Cosine function

$$f(\underline{x}) = \sum_{i=1}^{n} [\cos(-.05x_i + x_i^2)], \quad \underline{x}^{(0)} = [1, 1, ...1]^T$$

#### 4.3.4 Diagonal 5

$$f(\underline{x}) = \sum_{i=1}^{n} [ln(\text{exp}(x_i)) + \text{exp}(-x_i)], \quad \underline{x}^{(0)} = [1, 1, ...1]^T$$

#### 4.3.5 Diagonal 6

$$f(\underline{x}) = \sum_{i=1}^{n} [(\text{exp}(x_i)) + (1 - x_i)], \quad \underline{x}^{(0)} = [1, 1, ...1]^T$$

#### 4.3.6 Sine function

$$f(\underline{x}) = \sum_{i=1}^{n} [\sin(-.05x_i + x_i^2)], \quad \underline{x}^{(0)} = [1, 1, ...1]^T$$

#### 4.3.7 Diagonal 7

$$f(\underline{x}) = \sum_{i=1}^{n} [\text{exp}(x_i) - 2x_i - x_i^2], \quad \underline{x}^{(0)} = [1, 1, ...1]^T$$

#### 4.3.8 EG2

$$f(\underline{x}) = \sum_{i=1}^{n} [\sin(x_i + x_i^2 - 1) + \frac{1}{2}\sin(x_i^2)], \quad \underline{x}^{(0)} = [1, 1, ...1]^T$$

#### 4.3.9 Raydan 1

$$f(\underline{x}) = \sum_{i=1}^{n} \frac{1}{c}[\text{exp}(x_i) - x_i], \quad \underline{x}^{(0)} = [1, 1, ...1]^T$$

### 4.3.10 Diagonal 8

$$f(\underline{x}) = \sum_{i=1}^{n} [x_i \exp(x_i) - 2x_i - x_i^2], \quad \underline{x}^{(0)} = [1, 1, ...1]^T$$

## 4.4 Computational results

The CGMs Algorithm 4.1 was implemented with the direct method to solve all the numerical examples, and the results are presented in Table 3, while the same Algorithm was also implemented with the indirect method for eight different non-linear CGMs stated in Subsection 2.3 using MATLAB 1.8.0347 [R2009a] on a HP laptop computer 620 with processor Pentium (R) Dual-core CPU T4500 @2.30GB to solve five of the numerical examples, and their results were presented in Tables 4 to 11 using the following notations:
Itr (number of iterations), n (dimension), $f^*$ (optimal value of $f$), $\| g^* \|$ (norm of the optimal value of the gradient), CE (computational examples) and TET (total execution time in each CGM).

Table 3. Numerical solution for direct method.

| CE | n. (Dim) | Direct Method | | |
|---|---|---|---|---|
| | | Itr | $f^*$ | $\| g^* \|$ |
| 5.2.1 | 5000 | 1 | $5.0 * 10^3$ | $\underline{0}$ |
| | 10000 | 1 | $1.0 * 10^4$ | $\underline{0}$ |
| 5.2.2 | 5000 | 1 | $1.0217 * 10^3$ | $4.4865 * 10^{-8}$ |
| | 10000 | 1 | $-1.0435 * 10^3$ | $6.3448 * 10^{-8}$ |
| 5.2.3 | 5000 | 1 | $-5.0 * 10^3$ | $6.4404 * 10^{-8}$ |
| | 10000 | 1 | $-1.0 * 10^4$ | $9.1081 * 10^{-8}$ |
| 5.2.4 | 5000 | 1 | $5.0 * 10^3$ | $\underline{0}$ |
| | 10000 | 1 | $1.0 * 10^4$ | $\underline{0}$ |
| 5.2.5 | 5000 | 1 | $5.0 * 10^3$ | $\underline{0}$ |
| | 10000 | 1 | $1.0 * 10^4$ | $\underline{0}$ |
| 5.2.6 | 5000 | 1 | $5.0 * 10^3$ | $1.4755 * 10^{-7}$ |
| | 10000 | 1 | $1.0 * 10^4$ | $2.o866 * 10^{-8}$ |
| 5.2.7 | 5000 | 1 | $-4.0842 * 10^3$ | $2.4131 * 10^{-7}$ |
| | 10000 | 1 | $-8.1685 * 10^3$ | $3.4126 * 10^{-7}$ |
| 5.2.8 | 5000 | 1 | $2.5109 * 10^3$ | $3.1058 * 10^{-8}$ |
| | 10000 | 1 | $5.0217 * 10^3$ | $4.3923 * 10^{-8}$ |
| 5.2.9 | 5000 | 1 | $5.0 * 10^1$ | $\underline{0}$ |
| | 10000 | 1 | $1.0 * 10^2$ | $1.3408 * 10^{-7}$ |
| 5.2.10 | 5000 | 1 | $-2.4023 * 10^3$ | $\underline{0}$ |
| | 10000 | 1 | $-4.8045 * 10^3$ | $1.8962 * 10^{-7}$ |

## 4.5 Discussion on numerical results

From the tables of results, it can be observed that:

i: Both the direct and indirect methods produce accurate results and converge very fast.

ii: The eight nonlinear CGMs used for solving the computational examples gave the same results and utilized the same number of iterations although with slightly varying execution time.

iii: There were differences in the optimal value of $f$ obtained by both direct and indirect methods. The reason for this is as a result of the non-linearity of the problems giving rise to multiple solutions.

iv: In the Algorithm 4.1, the norm of $g^*$ is define as,

$$\| g^* \| = (\sum_{i=1}^{n} g_i^{*2}) < \epsilon \Rightarrow \sum_{i=1}^{n} g_i^{*2} < \epsilon^2.$$

In case every components of $g^*$ have the same value, at that point

$$ng_i^{*2} < \epsilon^2 \Rightarrow g_i^* < \sqrt{\frac{\epsilon^2}{n}} \forall i \quad (16).$$

So, substituting for the values of $n$ and $\epsilon$ in (16), then we have

$$g_i^* < \sqrt{\frac{10^{-12}}{10^4}} = 10^{-8} \forall i$$

so, if $g_k^* \neq 0$ and $g_i^* = 0 \forall i \neq k$, i.e.,

$$g_k^{*2} < \epsilon^2 \Rightarrow g_k^{*2} = \sqrt{\epsilon^2} = 10^{-6}.$$

Hence, $10^{-8} \leqslant g_i^* \leqslant 10^{-6}$, from which we conclude that $\| \underline{g}^* \| \approx \underline{0}$.

Table 4. Numerical solution for indirect method (TET=0.73).

| CGMs | n. (Dim) | BAN | | | Exc. Time |
|---|---|---|---|---|---|
| | | Itr | f* | Norm g* | |
| 5.2.1 | 5000 | 1 | 5.00e003 | 4.2e-012 | 0.02 |
| | 10000 | 1 | 1.00e.004 | 1.2e-011 | 0.02 |
| 5.2.4 | 5000 | 1 | 5.9375e003 | 7.2e-012 | 0.13 |
| | 10000 | 1 | 1.1875e.004 | 3.5e-012 | 0.07 |
| 5.2.5 | 5000 | 1 | 1.00e003 | 2.7e-011 | 0.08 |
| | 10000 | 1 | 2.00e.004 | 3.0e-011 | 0.06 |
| 5.2.7 | 5000 | 1 | -1.5089e004 | 2.4e-10 | 0.17 |
| | 10000 | 3 | -3.0179e.004 | 9.6e-10 | 0.07 |
| 5.2.9 | 5000 | 1 | 5.00e001 | 4.3e-013 | 0.04 |
| | 10000 | 1 | 1.00e.002 | 1.8e-013 | 0.07 |

Table 5. Numerical solution for indirect method (TET=0.52).

| CGMs | n. (Dim) | FR | | | Exc. Time |
|---|---|---|---|---|---|
| | | Itr | f* | Norm g* | |
| 5.2.1 | 5000 | 1 | 5.00e003 | 4.2e-012 | 0.02 |
| | 10000 | 1 | 1.00e.004 | 1.2e-011 | 0.02 |
| 5.2.4 | 5000 | 1 | 5.9375e003 | 7.2e-012 | 0.07 |
| | 10000 | 1 | 1.1875e.004 | 3.5e-012 | 0.07 |
| 5.2.5 | 5000 | 1 | 1.00e003 | 2.7e-011 | 0.10 |
| | 10000 | 1 | 2.00e.004 | 3.0e-011 | 0.05 |
| 5.2.7 | 5000 | 1 | -1.5089e004 | 2.4e-10 | 0.04 |
| | 10000 | 3 | -3.0179e.004 | 9.6e-10 | 0.06 |
| 5.2.9 | 5000 | 1 | 5.00e001 | 4.3e-013 | 0.04 |
| | 10000 | 1 | 1.00e.002 | 1.8e-013 | 0.05 |

Table 6. Numerical solution for indirect method (TET=0.46).

| CGMs | n. (Dim) | CD | | | Exc. Time |
|---|---|---|---|---|---|
| | | Itr | f* | Norm g* | |
| 5.2.1 | 5000 | 1 | 5.00e003 | 4.2e-012 | 0.02 |
| | 10000 | 1 | 1.00e.004 | 1.2e-011 | 0.02 |
| 5.2.4 | 5000 | 1 | 5.9375e003 | 7.2e-012 | 0.08 |
| | 10000 | 1 | 1.1875e.004 | 3.5e-012 | 0.06 |
| 5.2.5 | 5000 | 1 | 1.00e003 | 2.7e-011 | 0.03 |
| | 10000 | 1 | 2.00e.004 | 3.0e-011 | 0.05 |
| 5.2.7 | 5000 | 1 | -1.5089e004 | 2.4e-10 | 0.04 |
| | 10000 | 3 | -3.0179e.004 | 9.6e-10 | 0.06 |
| 5.2.9 | 5000 | 1 | 5.00e001 | 4.3e-013 | 0.04 |
| | 10000 | 1 | 1.00e.002 | 1.8e-013 | 0.06 |

Table 7. Numerical Solution for Indirect Method (TET=0.42).

| CGMs | n. (Dim) | DY | | | Exc. Time |
|---|---|---|---|---|---|
| | | Itr | f* | Norm g* | |
| 5.2.1 | 5000 | 1 | 5.00e003 | 4.2e-012 | 0.02 |
| | 10000 | 1 | 1.00e.004 | 1.2e-011 | 0.02 |
| 5.2.4 | 5000 | 1 | 5.9375e003 | 7.2e-012 | 0.05 |
| | 10000 | 1 | 1.1875e.004 | 3.5e-012 | 0.06 |
| 5.2.5 | 5000 | 1 | 1.00e003 | 2.7e-011 | 0.03 |
| | 10000 | 1 | 2.00e.004 | 3.0e-011 | 0.05 |
| 5.2.7 | 5000 | 1 | -1.5089e004 | 2.4e-10 | 0.04 |
| | 10000 | 3 | -3.0179e.004 | 9.6e-10 | 0.06 |
| 5.2.9 | 5000 | 1 | 5.00e001 | 4.3e-013 | 0.05 |
| | 10000 | 1 | 1.00e.002 | 1.8e-013 | 0.06 |

Table 8. Numerical solution for indirect method (TET=0.53).

| CGMs | n. (Dim) | HS | | | Exc. Time |
|---|---|---|---|---|---|
| | | Itr | f* | Norm g* | |
| 5.2.1 | 5000 | 1 | 5.00e003 | 4.2e-012 | 0.02 |
| | 10000 | 1 | 1.00e.004 | 1.2e-011 | 0.02 |
| 5.2.4 | 5000 | 1 | 5.9375e003 | 7.2e-012 | 0.05 |
| | 10000 | 1 | 1.1875e.004 | 3.5e-012 | 0.06 |
| 5.2.5 | 5000 | 1 | 1.00e003 | 2.7e-011 | 0.08 |
| | 10000 | 1 | 2.00e.004 | 3.0e-011 | 0.05 |
| 5.2.7 | 5000 | 1 | -1.5089e004 | 2.4e-10 | 0.07 |
| | 10000 | 3 | -3.0179e.004 | 9.6e-10 | 0.05 |
| 5.2.9 | 5000 | 1 | 5.00e001 | 4.3e-013 | 0.06 |
| | 10000 | 1 | 1.00e.002 | 1.8e-013 | 0.07 |

Table 9. Numerical solution for indirect method (0.47).

| CGMs | n. (Dim) | LS | | | Exc. Time |
|---|---|---|---|---|---|
| | | Itr | f* | Norm g* | |
| 5.2.1 | 5000 | 1 | 5.00e003 | 4.2e-012 | 0.02 |
| | 10000 | 1 | 1.00e.004 | 1.2e-011 | 0.02 |
| 5.2.4 | 5000 | 1 | 5.9375e003 | 7.2e-012 | 0.06 |
| | 10000 | 1 | 1.1875e.004 | 3.5e-012 | 0.07 |
| 5.2.5 | 5000 | 1 | 1.00e003 | 2.7e-011 | 0.04 |
| | 10000 | 1 | 2.00e.004 | 3.0e-011 | 0.07 |
| 5.2.7 | 5000 | 1 | -1.5089e004 | 2.4e-10 | 0.04 |
| | 10000 | 3 | -3.0179e.004 | 9.6e-10 | 0.06 |
| 5.2.9 | 5000 | 1 | 5.00e001 | 4.3e-013 | 0.04 |
| | 10000 | 1 | 1.00e.002 | 1.8e-013 | 0.05 |

Table 10. Numerical solution for indirect method (0.56).

| CGMs | n. (Dim) | PRP | | | Exc. Time |
|---|---|---|---|---|---|
| | | Itr | f* | Norm g* | |
| 5.2.1 | 5000 | 1 | 5.00e003 | 4.2e-012 | 0.02 |
| | 10000 | 1 | 1.00e.004 | 1.2e-011 | 0.02 |
| 5.2.4 | 5000 | 1 | 5.9375e003 | 7.2e-012 | 0.05 |
| | 10000 | 1 | 1.1875e.004 | 3.5e-012 | 0.07 |
| 5.2.5 | 5000 | 1 | 1.00e003 | 2.7e-011 | 0.08 |
| | 10000 | 1 | 2.00e.004 | 3.0e-011 | 0.07 |
| 5.2.7 | 5000 | 1 | -1.5089e004 | 2.4e-10 | 0.04 |
| | 10000 | 3 | -3.0179e.004 | 9.6e-10 | 0.05 |
| 5.2.9 | 5000 | 1 | 5.00e001 | 4.3e-013 | 0.04 |
| | 10000 | 1 | 1.00e.002 | 1.8e-013 | 0.12 |

Table 11. Numerical solution for indirect method (0.50).

| CGMs | n. (Dim) | HZ | | | Exc. Time |
|---|---|---|---|---|---|
| | | Itr | f* | Norm g* | |
| 5.2.1 | 5000 | 1 | 5.00e003 | 4.2e-012 | 0.02 |
| | 10000 | 1 | 1.00e.004 | 1.2e-011 | 0.02 |
| 5.2.4 | 5000 | 1 | 5.9375e003 | 7.2e-012 | 0.08 |
| | 10000 | 1 | 1.1875e.004 | 3.5e-012 | 0.07 |
| 5.2.5 | 5000 | 1 | 1.00e003 | 2.7e-011 | 0.04 |
| | 10000 | 1 | 2.00e.004 | 3.0e-011 | 0.07 |
| 5.2.7 | 5000 | 1 | -1.5089e004 | 2.4e-10 | 0.04 |
| | 10000 | 3 | -3.0179e.004 | 9.6e-10 | 0.06 |
| 5.2.9 | 5000 | 1 | 5.00e001 | 4.3e-013 | 0.04 |
| | 10000 | 1 | 1.00e.002 | 1.8e-013 | 0.06 |

## 5. Conclusion

In this research work, an effort has been made to discuss various numerical strategies available for solving unconstrained optimization problems of which CGM has been given more attention. More so, two different methods were discussed for minimizing non-polynomial unconstrained optimization problems with optimum line search technique using eight different nonlinear CGMs. The direct and indirect methods were implemented on some test functions, the result of which were given in Tables 1 to 11.

## References

[1] K. Amini and P. Faramarzi and N. Pirfalah, A modified Hestenes-Stiefel conjugate gradient method with an optimal property, Optimization Methods and Software, **34 (4)** (2019) 770–782.

[2] N. Andrei, Unconstrained optimization test functions unpublished manuscript, Advanced Modeling and Optimization, **10** (2008) 147–161.

[3] S. Babaie-Kafaki, On the sufficient descent condition of the Hager-Zhang conjugate gradient methods, 4OR, **12**(2014) 285–292.

[4] S. Babaie-Kafaki, On the sufficient descent property of the Shannos conjugate gradient method, Optimization Letters, **7 (4)** (2013) 831–837.

[5] O. M. Bamigbola and M. M. Ali and E. Nwaeze, An efficient and convergent method for unconstrained nonlinear optimization, International Congress of Mathematicians, Hyderabad, India, (2010).

[6] Y. Dai and Y. Yuan, A nonlinear conjugate gradient with strong global convergence properties, SIAM Journal on Optimization, **10** (2000) 177–182.

[7] W. C. Davidon, Variable Metric Method of Minimization, Argonne National Laboratory Report, (1959).

[8] R. Fletcher, Practical method of Optimization, Second edition, John Wiley, New York, (1987).

[9] R. Fletcher and M. J. D. Powell, A Rapidly Convergent Descent Method for Minimization, Computer Journal, **6 (2)** (1963) 163–168.

[10] R. Fletcher and C. M. Reeves, Function minimization by conjugate gradients, Computer Journal, **7** (1964) 149–154.

[11] W. W. Hager and H. Zhang, A new conjugate gradient method with guaranteed descent and an effiecient line search, SIAM Journal on Optimization, **16** (2005) 170–172.

[12] M. R. Hestenes and E. Stiefel, Methods of conjugate gradient for solving linear equations, Journal of research of the National Bureau of Standards, **49** (1952) 409–436.

[13] A. A. Ishaq, T. Latunde and K. B. Akande, A step-length for conjugate gradient methods, Malaysian Journal of Computing, **5 (1)** (2020) 403–413.

[14] H. Jinhong and Z. Genjiao, A conjugate gradient method without line search and the convergence analysis, In: Fourth International Conference on Emerging Intelligent Data and Web Technologies, Granzhoul, Jiangxi 341000, China, (2013) 37–86.

[15] H. Kobayashi and Y. Narushima and H. Yabe, Descent three-term conjugate gradient methods based on secant conditions for unconstrained optimization, Optimization Methods and Software, **32 (6)** (2017) 1313–1329.

[16] Y. Liu and C. Storey, Efficient generalized conjugate gradient algorithms, Journal of Optimization Theory and Application, **69** (1992) 129–137.

[17] P. Mtagulwa and P. Kaelo, A convergent modified HS-DY hybrid conjugate gradient method for unconstrained optimization problems, Journal of Information and Optimization Sciences, **40 (1)** (2019) 97–113.

[18] L. Muhammad and M. Y. Waziri, An alternative three-term conjugate gradient method algorithm for systems of non-linear equations, International Journal of Mathematical Modeling and Computations, **7 (2)** (2017) 145–157.

[19] E. Polak and G. Ribiere, Note sur la convergence de directions conjugees, Rev. Francaise Informat Recherche Operationelle, **16** (1969) 35–43.

[20] S. S. Rao, Engineering Optimization, Theory and Practice, Fourth Edition, John Wiley and Sons Inc., New York, (2009).

[21] M. L. Robert and T. Virginia and W. T. Micheal, A sufficient descent LiuStorey conjugate gradient method and its global convergence, Journal of Computational and Applied Mathematics, **124 (1-2)** (2000) 191–207.

[22] A. Z. M. Sofi, M. Mamat and I. Mohd, An improved BFGS search using exact line search for solving unconstrained optimization problems, Journal of Mathematical Sciences, **7** (2013) 73–85.

[23] Z. Sun, H. Li, J. Wang and Y. Tian, Two modified spectral conjugate gradient methods and their global convergence for unconstrained optimization, International Journal of Computer Mathematics, **95 (10)** (2018) 2082–2099.

[24] N. Wataru, N. Yasushi and Y. Hiroshi, Nonlinear conjugate gradient methods with sufficient descent properties for unconstrained optimization, Journal of Industrial and Management Optimization, **9** (2013), 595–619.

[25] T. J. Ypma, Historical development of the Newton-Raphson method, Society for Industrial and Applied Mathematics, **37** (1995) 521–551.

[26] G. Yuan, B. Wang and Z. Sheng, The Hager-Zhang conjugate gradient algorithm for large-scale nonlinear equations, International Journal of Computer Mathematics, **96 (8)** (2019) 1533–1547.

[27] U. K. M. Yusof, M. A. H. Ibrahim, M. Rivaie, M. Mamat, M. A. Mohammed and P. L. Ghazali, Hybrid quasi-Newton with new conjugate gradient method using exact line search, International Journal of Recent Technology and Engineering, **7** (2019) 651–655.

[28] B. Zhang, Convergence of modified conjugate gradient methods without line search, Second International Conference on Computational Intelligence and Natural Computing (CINC), Feixian School, Linyi Normal University, China, (2010).

[29] K. Zhang and H. Liu and Z. Liu, A Dai-Liao conjugate gradient method with optimal parameter choice, Numerical Functional Analysis and Optimization, **40 (2)** (2019) 194–215.