# Numerical Experiments of the Comjugate Gradient Method with and without Line Search

**A. Ajimoti and O. M. Bamigbola**
Department of Mathematics,
Faculty of Physical Sciences,
University of Ilorin, Ilorin,
Nigeria.
Email: kadam4best@gmail.com

***ABSTRACT***

*The conjugate gradient method CGM is an effective iterative method which is widely used for solving large-scale unconstrained optimization problems due to its low memory requirement. The efficiency of the CGM depends majorly on the step-size. Line search technique has been used in various literatures to obtain the step-size. A very recent development is to obtain the step-size with a unified formula which is refereed to as step-size without line search. Hence, in this work, we present numerical experiments for well-known CGMs such as Fletcher-Reeves, Bamigbola-Ali-Nwaeze, Polak-Ribiere, Dai-Yuan, Liu-Storey, Hesten-Stiefel, Conjugate-Descent, Hager-Zhang and Gradient Search Conjugacy methods. Numerical results obtained are graphically illustrated using performance profiling software to compare numerical efficiency of five inexact line searches namely Armijo, Goldstein, Weak, Strong and Approximate Wolfe and two formulae for estimating the step-size without line search which are Wu formula and Ajimoti-Bamigbola formula.*

## 1.1 INTRODUCTION

Consider the conjugate gradient method for unconstrained optimization problem

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

where $\Re^n$ is an n-dimensional Euclidean vector space and $f:\Re^n \to \Re$ is a real valued, continuously differentiable function. The CGM is an effective iterative method which is widely used due to its very low memory requirement and fast convergence ability for engineers and mathematicians who are interested in solving large-scale problems (Jinhong and Genjiao, 2013). A CGM generates a set of quantities $x^{(k)}$, $\alpha_k$, $d^{(k)}$, $\beta_k$ and $g^{(k)}$ at iteration $k$, where $x^{(k)}$ is the $k^{th}$ iterate solution, $\alpha_k$ is a positive step size obtained by a step size rule, $d^{(k)}$ is the search direction, $\beta_k$ is the conjugate gradient parameter which determines the different conjugate gradient methods which have different numerical effects (Zhang, 2010) and $g^{(k)}$ denotes the gradient of $f(x)$ $(i.e., \nabla f(x^{(k)}))$ at the point $x^{(k)}$. The different conjugate gradient methods differ in the way of selecting $\beta_k$. Some well-known formulae for $\beta_k$ are the following

$$\beta_k^{FR} = \frac{\left\|g^{(k+1)}\right\|^2}{\left\|g^{(k)}\right\|^2}, \text{Fletcher and Reeves (1964)[FR]} \tag{2}$$

$$\beta_k^{PRP} = \frac{g^{(k+1)T} y^{(k)}}{\left\| g^{(k)} \right\|^2}, \text{Polak and Ribiere (1969)[PRP]} \tag{3}$$

$$\beta_k^{HS} = \frac{g^{(k+1)T} Y^{(k)}}{g^{(k)T} g^{(k)}}, \text{Hestenes and Stiefel (1952)[HS]} \tag{4}$$

$$\beta_k^{CD} = -\frac{\left\| g^{(k+1)} \right\|^2}{g^{(k)T} d^{(k)}}, \text{Fletcher (1987)[CD]} \tag{5}$$

$$\beta_k^{DY} = \frac{\left\| g^{(k+1)} \right\|^2}{y^{(k)T} d^{(k)}}, \text{Dai and Y.Yuan (2000)[DY]} \tag{6}$$

$$\beta_k^{LS} = -\frac{g^{(k+1)T} y^{(k)}}{g^{(k)T} d^{(k)}}, \text{Liu and Storey (1992)[LS]} \tag{7}$$

$$\beta_k^{BAN} = \frac{g^{(k+1)T} y^{(k)}}{g^{(k)T} y^{(k)}}, \text{Bamigbola et al. (2010)[BAN]} \tag{8}$$

$$\beta_k^{HZ} = \left( y^{(k)} - 2d^{(k)} \frac{\left\| y^{(k)} \right\|^2}{y^{(k)T} d^{(k)}} \right)^{(T)} \frac{g^{(k+1)T} d^{(k)}}{y^{(k)T} d^{(k)}}, \text{Hager and Zhang (2005)[HZ]} \tag{9}$$

$$\beta_k^{GSC} = \frac{g^{(k+1)T} g^{(k)}}{g^{(k)T} d^{(k)}}, (Gradient Search Conjugacy)[GSC] \tag{10}$$

where $y^{(k)} = (g^{(k+1)} - g^{(k)})$.

In general, the conjugate gradient method uses an iterative scheme of the form:

$$x^{(k+1)} = x^{(k)} + \alpha_k d^{(k)} \tag{11}$$

where, $x^{(0)}$ is the initial iterative point, the step size $\alpha_k$ is positive which can be determined by various step-size rules and the directions $d^{(k)}$ are generated by the rule

$$d^{(k)} = \begin{cases} -g^{(k)} + \beta_{(k-1)} d^{(k-1)}; k \geq 1 \\ -g^{(k)}; \quad k = 0 \end{cases} \tag{12}$$

## 2 CGMs With and Without Line Search

The basic philosophy of most numerical methods is to produce a sequence of improved approximations to the optimum according to the following scheme:

Step 1: Choose an initial trial point $x^{(0)} \in \Re^n$.

Step 2: Determine if $\|g^{(0)}\| = 0$, then stop, otherwise, proceed to step 3.

Step 3: Find a suitable direction $d^{(k)}$ which points in general direction of the optimum such that $g^{(k)T} d^{(k)} < 0$,

Step 4: find an appropriate step size $\alpha_k \geq 0$ for movement along the direction $d^{(k)}$.

Step 5: obtain the new approximation $x^{(k+1)}$ as $x^{(k+1)} = x^{(k)} + \alpha_k d^{(k)}$.

Step 6: Test if $x^{(k+1)}$ is the optimum.

The iterative procedure indicated by step 5 above is valid for both constrained and unconstrained optimization. In fact from step 5 the success of an optimization method is largely dependent on the accuracy of computing the search direction $d^{(k)}$ and the step size $\alpha_k$.

In performing the iterative scheme in step 5, various methods are applicable in determining the step-size $\alpha_k$ one of such methods is the line search technique. On the other hand, a fixed formula can also be used to obtain the step-size $\alpha_k$ in step 5 above which is simply referred to as step-size without line search (Jie and Jiapu, 2001).

### 2.1 Line Search

This is an important step in the conjugate gradient method algorithm when solving unconstrained optimization problems. In every line search, the choice of technique for determining $\alpha_k$ affects both the convergence and the speed of convergence of the algorithm. The two types of line search procedures basically in use are Exact Line Search and Inexact Line Search.

#### 2.1.1 Exact Line Search

The aim of every line search is to determine $\alpha_k \geq 0$ along the search direction $d^{(k)}$ with the objective of ensuring a non-deteriorating rate of convergence. Then to achieve this, we first set $\alpha_k = \alpha^*$, such that,

$$\alpha^* = argmin_{\alpha \geq 0} f(x^{(k)} + \alpha d^{(k)}) \tag{13}$$

i.e., $\alpha_k$ is the value of $\alpha_k \geq 0$ which minimizes the function $f$ along $d^{(k)}$. Thus, $\alpha^*$ in (19) can be obtained by solving the differential equation,

$$\frac{d}{d_\alpha} f(x^{(k)} + \alpha d^{(k)}) = 0 \tag{14}$$

The technique employed in (14) yields an exact value $\alpha^*$ and is referred to as an exact line search. However, commonly, the exact line search is cost expensive, especially when an iterate is far from the solution of the problem (Sun and Yuan, 2006).

### 2.1.2 Inexact Line Search

It is important to note that exact line search is expensive to carryout, as result of the limitation of the exact line search there is a need for line search which can identify a step-size that produces substantial reduction in the value of objective function at minimal cost. In a case of nonlinear unconstrained optimization problems, inexact line search technique is cost efficient and more precise to work with. The frame work of inexact line search is as follows:

- Generate a criterion that ensures the step-size $\alpha$ is neither too long nor too short.
- Pick a good initial step-size to kick start the algorithm.
- Construct a sequence of updates that satisfy the criterion formulated in (i) after every few steps.

There are various rules for accepting a step-size, some of which are:

- Goldstein (Goldstein, 1965).
- Armijo (Armijo, 1966).
- Wolfe (Wolfe, 1969), (Wolfe, 1971).
- Powel (Powell, 1976).
- Luenberger (Luenberger, 1984).
- Fletcher (Fletcher, 1987).
- Boggs and Schnabel (Boggs and Schnabel, 1987).
- Gill et al. (Gill et al., 1982).
- Hager and Zhang (Hager and Zhang, 2005), and so on.

**Choice of 5 inexact Line Search**

**1. Armijo Line Search Criterion**

This is the simplest among the inexact line search procedures and was introduced by (Armijo, 1966). The Armijo rule works in such a way that it first guarantees that the selected step-length $\alpha_k$ is not too large, and then that it is not too small.

Consider the function

$$\phi(\alpha_k) = f(x^{(k)} + \alpha_k d^{(k)}) \tag{15}$$

A step-length $\alpha_k$ satisfies the Armijo's rule if

$$\phi(\alpha_k) \le \phi(0) + \sigma \nabla \phi(0) \alpha_k \tag{16}$$

$$\phi(\alpha_k \eta) \ge \phi(0) + \sigma \nabla \phi(0) \alpha_k \eta \tag{17}$$

The first condition ensures that $\alpha_k$ is not too large while the second one makes it not to be too small.

The constants $\sigma$ and $\eta$ must be chosen such that $0 \le \sigma \le 1$ and $\eta \ge 1$

**2. Goldstein Line Search Criteria**

The Goldstein line search, first introduced by A. A. Goldstein (Goldstein, 1965) accepts a step-length $\alpha_k \geq 0 > 0$ if it satisfies the conditions:

$$\delta_1 \alpha_k g^{(k)T} d^{(k)} \leq f(x^{(k)} + \alpha_k d^{(k)}) - f(x^{(k)}) \leq \delta_2 \alpha_k g^{(k)T} d^{(k)} \quad (18)$$

where $0 < \delta_1 < \frac{1}{2} < \delta_2 < 1$

### 3. Wolfe Line Search Criteria

This was first published by Philip Wolfe (Wolfe, 1969). He proposed that step-length $\alpha_k$ is considered optimal under the Wolfe line search if it satisfies the two Wolfe Conditions

- $$f(x^{(k)} + \alpha_k d^{(k)}) \leq f(x^{(k)}) + \sigma_1 \alpha_k g^{(k)T} d^{(k)} \quad (19)$$
- $$g(x^{(k)} + \alpha_k d^{(k)})^T \geq \sigma_2 g^{(k)T} d^k \quad (20)$$

and $\phi(\alpha_k) = f(x^{(k)} + \alpha_k d^{(k)})$ from where $0 \leq \sigma_1 \leq \sigma_2 \leq 1$. The first inequality ensures that the function reduced sufficiently, and the second prevents the steps from being too small.

However, in 1971, he notified in (Wolfe, 1971) that there are cases where a step-size may satisfy the general Wolfe condition without necessarily minimizing the function $\phi(\alpha_k)$. As such, a more strict two sided test is placed on the gradient of $\phi$. This forces $\alpha_k$ to lie at last in the neighborhood of a local minimizer of $\phi$. This test is called the strong Wolfe conditions.

### 4. Strong Wolfe Conditions

$$f(x^{(k)} + \alpha_k d^{(k)}) \leq f(x^{(k)}) + \sigma_1 \alpha_k g^{(k)T} d^{(k)} \quad (21)$$

$$| g(x^{(k)} + \alpha_k d^{(k)})^T | \leq \sigma_2 | g^{(k)T} d^{(k)} | \quad (22)$$

$$0 \leq \sigma_1 \leq \sigma_2 \leq 1$$

### 5. Approximate Wolfe Line Search Criteria

The Approximate Wolfe line search was introduced by Haggar and Zhang, (Hager and Zhang, 2005). This line search accepts any step-length $\alpha_k > 0$ if and only if it satisfies the the conditions:

$$(2\delta - 1)\phi'(0) \geq \phi'(\alpha_k) \geq \rho\phi'(0) \quad (23)$$

where $\phi(\alpha_k) = f(x^{(k)} + \alpha_k d^{(k)})$ and $0 < \delta < \frac{1}{2} < \rho < 1$

### 2.2 CGM Without Line Search

When the step-size $\alpha \geq 0$ is obtained by a unified formula rather than line search process, then it is referred to as step-size without line search. Shun and Zhang (Jie and Jiapu,

2001) developed a CGM where the step-size in their method is computed by a formula rather than a line search technique. The formula is given by:

$$\alpha_k = \frac{-\delta g^{(k)} d^{(k)}}{\left\| d^{(k)} \right\|^2_{Q_{(k)}}} \quad (24)$$

where $\left\| d^{(k)} \right\|_{Q_{(k)}} = \sqrt{d^{(k)T} Q_{(k)} d^{(k)}}$ $\delta \in \left(0, \frac{V_{min}}{\tau}\right)$, $\tau$ is a Lipschitz constant of $f$ and $\{Q_{(k)}\}$ is a sequence of positive definite matrices satisfying for positive constant $V_{min}$ and $V_{max}$ that

$$V_{min} d^T d \le d^T Q_{(k)} d \le V_{max} d^T d, d \in \Re^n \quad (25)$$

They used the unique formula for $\alpha_k$ in (25) to prove the global convergence for five kinds of conjugate gradient methods. Chen and Sun (Xiongda and Jie, 2002) proved that the same formula for $\alpha_k$ ensured global convergence for a two-parameter family of conjugate gradient methods. But the formula for the step-size $\alpha_k$ above involve a positive matrix $\{Q_{(k)}\}$. For large-scale optimization problems, this may cost additional memory space and execution time during the computations. Later, Qing-jun Wu (Wu, 2011) derived a formula for the step-size $\alpha_k$ that is matrix free and uses both available function and gradient information. He updated the formula in equation (24) above based on the quasi-Newton methods in (Zhang et al., 1999) and (Zhang and Xu, 2001), the formula for $\alpha_k$ is presented as

$$\alpha_k = \frac{-\delta g^{(k)T} d^{(k)}}{(\bar{g}^{(k+1)} + g^{(k)})^T d^{(k)} + \gamma \theta_k} \quad (26)$$

where $\theta_k = 6(f^{(k)} - \bar{f}^{(k+1)}) + 3(g^{(k)} - \bar{g}^{(k+1)})^T d^{(k)}$, $\delta$ and $\gamma$ are parameter satisfying, $\delta \in (0, \frac{K}{\tau})$ and $\gamma \ge 0$ if $\tau = K$.

Recently, (Ajimoti and Bamigbola, 2016) derived a new formula to obtain the step-size $\alpha_k$ without any line search processes which does not contain any matrix computation and uses only gradient information to obtain the step-size. The formula is presented below:

$$\alpha_k = \frac{-\delta g^{(k)T} d^{(k)}}{(\bar{g}^{(k+1)} - g^{(k)})^T d^{(k)}}, \delta \in (0, \frac{\tau}{\lambda}) \quad (27)$$

### 2.3 ALGORITHM : CGM Algorithm

- Select the initial point, $x^{(0)} \in \mathrm{R}^n$, $\varepsilon \ge 0$ (a small number called tolerance) and set $d^{(0)} = -g^{(0)} = -\nabla f(x^{(0)}), k = 0$.
- Terminate process if $\| g^{(0)} \| \le \varepsilon$, otherwise, go to Step 3.
- Compute step length $\alpha_k$ by:
- Using fixed formula in (26).
- Using fixed formula in (27).

• Using 5 inexact line search in subsection 2.1.2.

• Set $x^{(k+1)} = x^{(k)} + \alpha_k d^{(k)}$; if $\| g^{(k+1)} \| \leq \varepsilon$, then stop, otherwise, go to the next Step.

• Compute the search direction $d^{(k+1)} = -g^{(k+1)} + \beta_k d^{(k)}$. Where $\beta_k$ is given by equation $(2-10)$.

• Set $k = k+1$, and go to step 3.

## 3 COMPUTATIONAL CONSIDERATION

The computational experiments carried out in this research incorporated with fixed formulae in (26), (27) and 5 choices of inexact line search in subsection 2.1.2 into the CGM Algorithm 2.3. Our aim is to perform an experiment that aid in measuring the effectiveness of various step-size rules for obtaining the step size $\alpha_k$ either by fixed formulae or line search procedures. The formulae for finding step size without line search were compared with five other conventional inexact line searches rules by using nine kinds of conjugate gradient methods to solve thirty unconstrained optimization test functions obtained from the CUTE collection made available by Andrei (2008) and Jamil and Yang (2013) with standard starting points and each test function is given with two different dimensions ($n = 5{,}000$ and $n = 10{,}000$). Hence a total number of 3780 computations for the seven step-size solvers against nine types of CGMs were considered. The Algorithm 2.3 was implemented via Matlab 8.0 version and run on a PC HP EliteBook 6930p with 2.00GB RAM memory, 2.20GHZ processor and 3.4 windows experience index operating system. In figures $1-14$, we adopt the performance profiles introduced by Dolan and More (2002) to evaluate and compare the performance of different kinds of CGM against various rules of obtaining the step-size $\alpha_k$, to test the efficiency of these methods using optimization software based on the CPU time and the number of iteration where ABR represents Ajimoti-Bamigbola rule for step-size without line search, WR represents Wu rule for step-size without line search, ARR represents Armijo line search rule, SWR represents strong Wolfe line search rule, WWR represents weak Wolfe line search rule, GR represents Goldstein line search rule and AWR represents approximate Wolfe line search rule.

A table containing the test functions and their sources is presented in Table 1 and their numerical results are graphically illustrated in figures $1-14$.

Table 1: A list of test problems

| S/N | PROBLEM | SOURCE |
|---|---|---|
| 1 | Extended Rosenbrock Function | (Andrei, 2008) |
| 2 | Linear Function - rank 1 | (Andrei, 2008) |
| 3 | Quadratic Diagonal Perturbed Function | (Andrei, 2008) |

| | | |
|---|---|---|
| 4 | Perturbed Quadratic Function | (Andrei, 2008) |
| 5 | Quadratic QF1 Function | (Andrei, 2008) |
| 6 | ARGLINB(m=20) Function | (Andrei, 2008) |
| 7 | Almost Perturbed Quadratic Function | (Andrei, 2008) |
| 8 | Extended White & Holst Function | (Andrei, 2008) |
| 9 | Raydan 1 Function | (Andrei, 2008) |
| 10 | Raydan 2 Function | (Andrei, 2008) |
| 11 | Extended Three Exponential Terms Function | (Andrei, 2008) |
| 12 | Generalized Rosenbrock Function | (Andrei, 2008) |
| 13 | Generalized White & Holst Function | (Andrei, 2008) |
| 14 | Extended Block Diadonal BD1 Function | (Andrei, 2008) |
| 15 | HimmelBG Function | (Andrei, 2008) |
| 16 | Power Function | (Andrei, 2008) |
| 17 | Extended Dixon and Price | (Andrei, 2008) |
| 18 | Extended Booth Function | (Andrei, 2008) |
| 19 | Extended Boh2 Function | (Andrei, 2008) |
| 20 | Diagonal 3 Function | (Jamil and Yang, 2013) |
| 21 | Hager Function | (Jamil and Yang, 2013) |
| 22 | Extended Penalty Function | (Andrei, 2008) |
| 23 | Extended Cliff & Roth Function | (Andrei, 2008) |
| 24 | Extended Quadratic Penalty QP1 Function | (Andrei, 2008) |
| 25 | Extended EP1 Function | (Andrei, 2008)] |
| 26 | ARWHEAD Function | |

| | | |
|---|---|---|
| | | (Andrei, 2008) |
| 27 | Extended Freudenstein & Roth Function | (Andrei, 2008) |
| 28 | Cube Function | (Andrei, 2008) |
| 29 | Extended Goldstein & Price Function | (Andrei, 2008) |
| 30 | Chebyquad Function | (Andrei, 2008) |

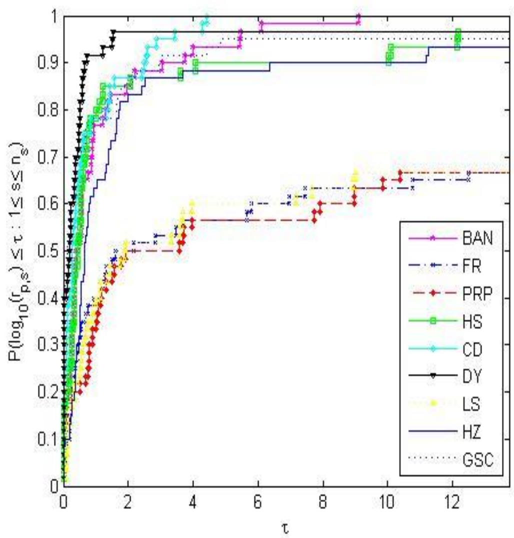


Figure 1: Performance profile for various CGMs by ABR based on CPU time

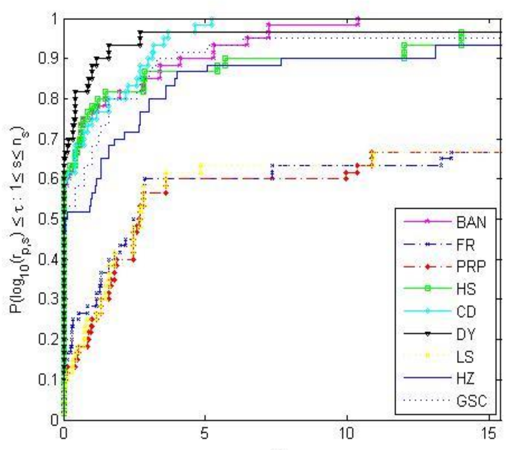


Figure 2: Performance profile for various CGMs by ABR based on number of iterations

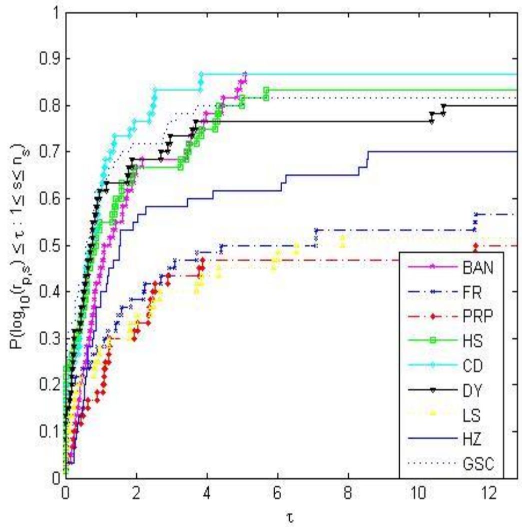


Figure 3: Performance profile for various CGMs by WR based on CPU time

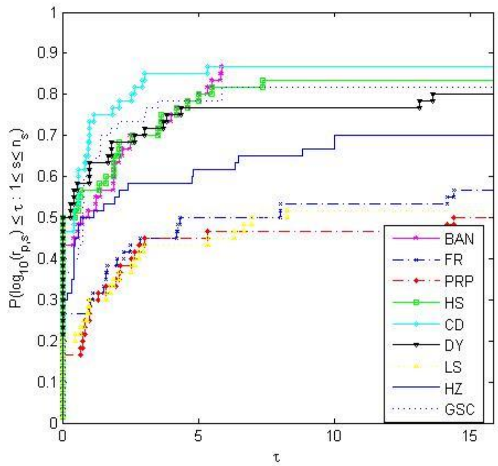


Figure 4: Performance profile for various CGMs by WR based on number of iterations

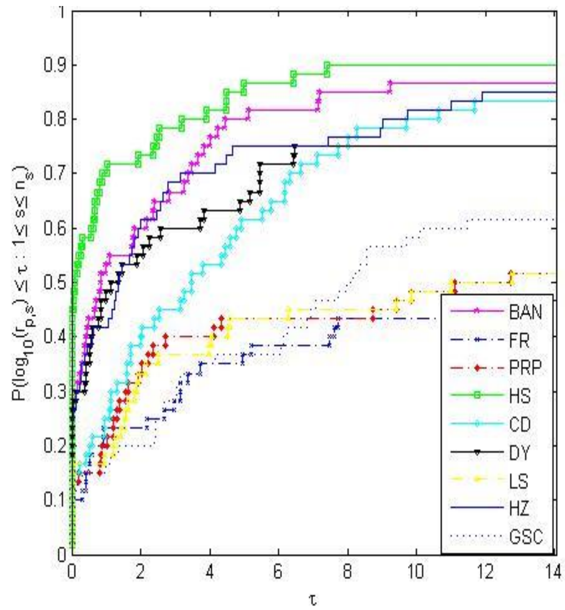


Figure 5: Performance profile for various CGMs by ARR based on CPU time

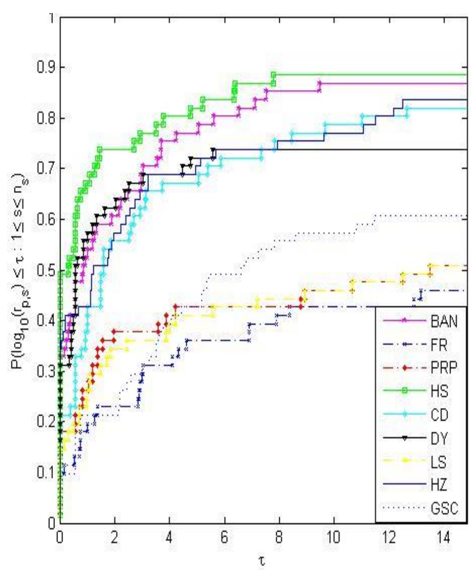


Figure 6: Performance profile for various CGMs by ARR based on number of iterations

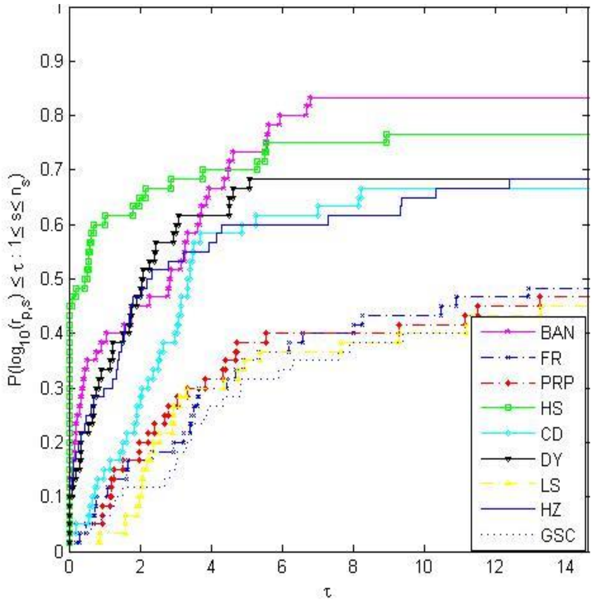


Figure 7: Performance profile for various CGMs by SWR based on CPU time

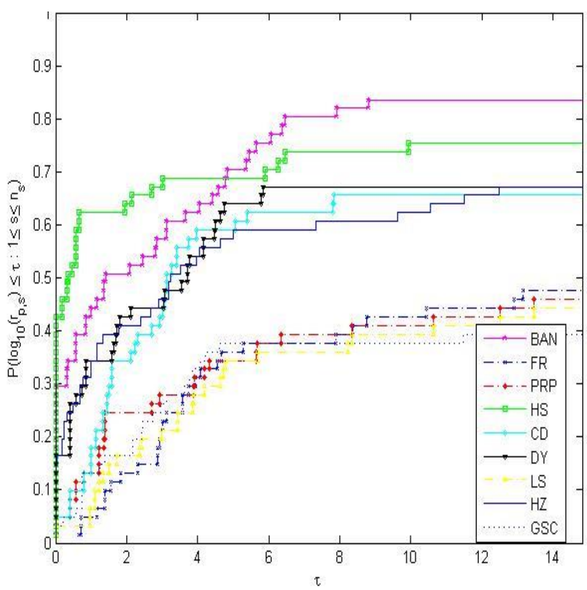


Figure 8: Performance profile for various CGMs by SWR based on number of iterations

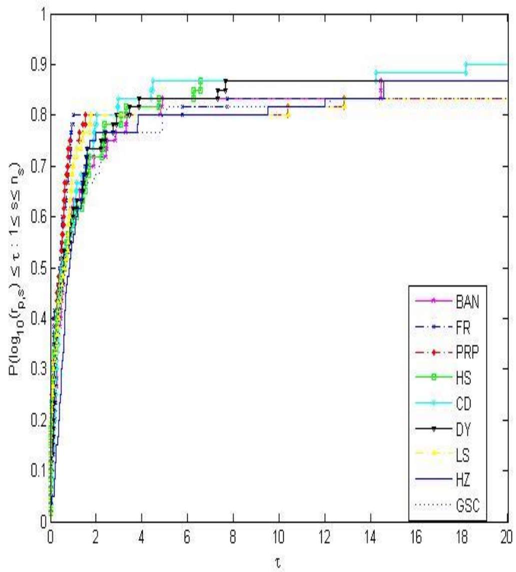


Figure 9: Performance profile for various CGMs by WWR based on CPU time

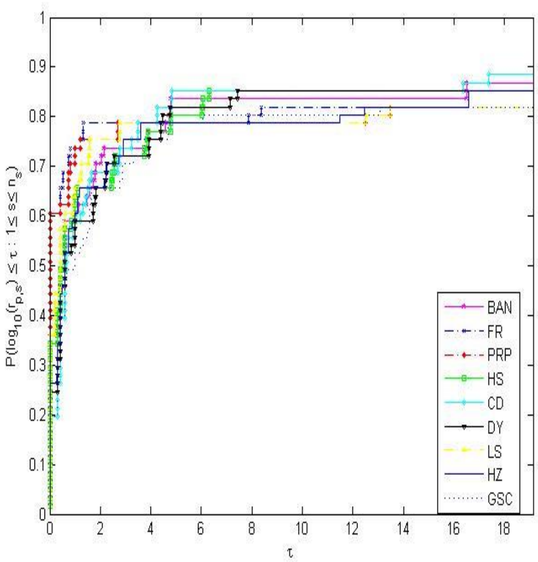


Figure 10: Performance profile for various CGMs by WWR based on number of iterations

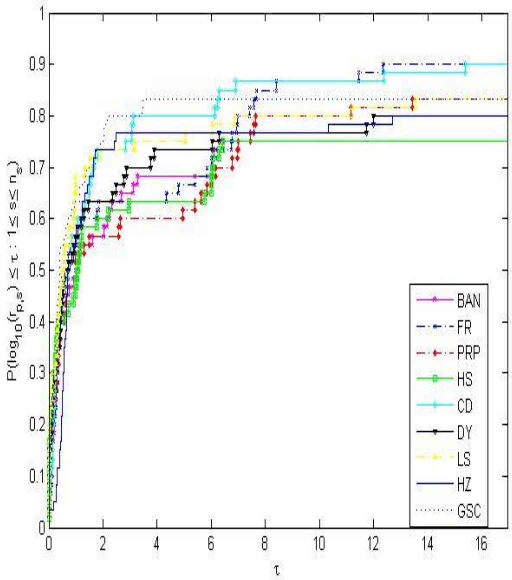


Figure 11: Performance profile for various CGMs by GR based on CPU time

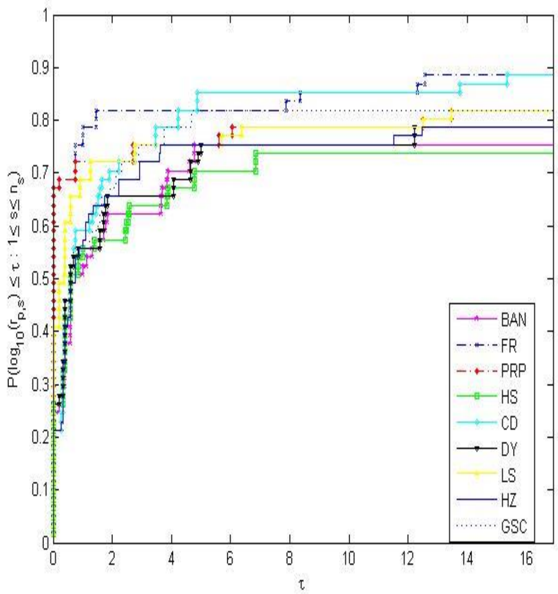


Figure 12: Performance profile for various CGMs by GR based on number of iterations

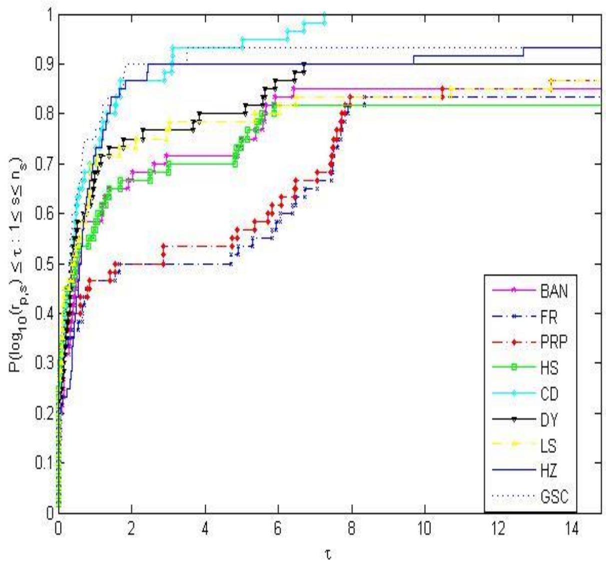


Figure 13: Performance profile various CGMs by AWR based on CPU time

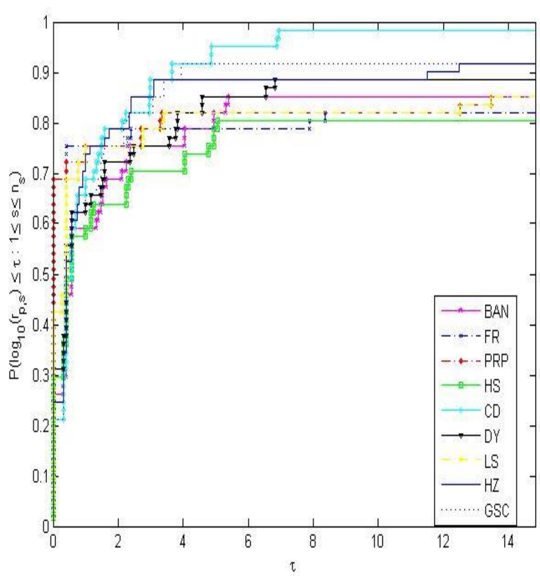


Figure 14: Performance profile for various CGMs by AWR based on number of iterations

### 3.1 Remarks on Computational Results

**Solverbility Index**

For better description and understanding of the Figures above, the solvability measure for the CG methods are presented below based on the number of successes and failures in percentage recorded by each Step-Size Rules $(SSR)$:

Table 2: Solvability index for different SSR by CG methods

| SSR | Soverbility | Nonlinear CG Method | | | | | | | | | AVERAGE |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | Index | BAN | FR | PRP | HS | CD | DY | LS | HZ | GSC | SUCC./FAIL. |
| ABR | Success | 100 | 67 | 67 | 97 | 100 | 97 | 67 | 94 | 97 | 88.2 |
| | Failure | 00 | 33 | 33 | 03 | 00 | 03 | 33 | 06 | 03 | 11.8 |
| WR | Success | 87 | 58 | 50 | 84 | 87 | 80 | 52 | 70 | 82 | 72.2 |
| | Failure | 13 | 42 | 50 | 16 | 13 | 20 | 48 | 30 | 18 | 27.8 |
| ARR | Success | 87 | 47 | 52 | 90 | 83 | 75 | 52 | 85 | 62 | 70.3 |
| | Failure | 13 | 53 | 48 | 10 | 17 | 25 | 48 | 15 | 38 | 29.7 |
| SWR | Success | 83 | 48 | 47 | 77 | 67 | 68 | 45 | 68 | 40 | 60.3 |
| | Failure | 17 | 52 | 53 | 23 | 33 | 32 | 55 | 32 | 60 | 39.4 |
| WWR | Success | 87 | 84 | 83 | 87 | 90 | 87 | 83 | 87 | 84 | 85.8 |
| | Failure | 13 | 16 | 17 | 13 | 10 | 13 | 17 | 13 | 16 | 14.2 |
| GR | Success | 76 | 90 | 83 | 75 | 90 | 80 | 83 | 80 | 84 | 82.3 |
| | Failure | 24 | 10 | 17 | 25 | 10 | 20 | 17 | 20 | 16 | 17.7 |
| AWR | Success | 85 | 84 | 87 | 82 | 100 | 90 | 87 | 94 | 93 | 87.4 |
| | Failure | 15 | 16 | 13 | 18 | 00 | 10 | 13 | 06 | 06 | 12.6 |
| AVERAGE | Success | 86.4 | 68.4 | 67.1 | 84.7 | 88.1 | 82.6 | 67.1 | 82.9 | 77.4 | |
| | Failure | 13.6 | 31.6 | 32.9 | 15.3 | 11.9 | 17.4 | 32.9 | 17.1 | 22.6 | |

From Table 2 which measures the effectiveness of each CGMs and the various SSRs presented in our work. It is evident that BAN and CD methods successfully solved more problems while PRP and FR gave a least number of success among the CGMs considered. The step-size rules $(SSR)$ for without line search techniques used in this work are ABR and WR. The ABR showed a superior efficient performance displaced over the WR. It is noticed that the step-size rules with line search procedures considered in this research work. AWR achieved strong efficient performance compared to the other inexact line searches while ARR and SWR were lagging behind.

It is observed that despite the inefficiency of FR and PRP methods, they both are very efficient using GR and AWR respectively. It is also observed that almost all the CG method considered attained the highest level of efficiency using the Ajimoti-Bamigbola rule without line search to obtain the step-size and it is evident to finalize that ABR showed better efficient performance displaced over AWR judged by the calculated average as presented in the table above.

## 4 Conclusion

In this research work, it is established that among all the step-size rules used, the ABR for finding the step without any line search procedure was able to obtain the step-size faster and guarantee solution to the CGMs through the computational results most especially with the BAN and CD methods. It is also established that the BAN method and Conjugate-Descent method exhibit better efficiency when compared to the other conjugate gradient methods mentioned in this paper.

**Acknowledgments**

The authors are very grateful to anonymous referees and editors for careful reading and valuable suggestions and comments that greatly improved the paper.

**References**

Ajimoti A. and Bamigbola, O. M. (2016). A gradient based step-size for conjugate gradient methods. Proceeding of 35th Annual Conference of the Nigeria Mathematical Society.

Andrei, N. (2008). Unconstrained optimization test functions unpublished manuscript. Advanced Modeling and Optimization, Vol. 10:pp. 147{161.

Armijo, L. (1966). Minimization of functions having lipschitz continuous first partial derivatives. Pacific Journal of Mathematics, Vol. 16:pp. 1{3.19

Bamigbola, O. M., Ali, M. M., and Nwaeze, E. (2010). An efficient and convergent method for unconstrained nonlinear optimization. A paper presented at the International Congress of Mathematicians, Hyderabad, India.

Boggs, P. T. and Schnabel, R. B. (1987). Numerical optimization. Lecture Note for a Short Course.

Dai, Y. and Y.Yuan (2000). A nonlinear conjugate gradient with strong global convergence properties. SIAM Journal on optimization, Vol. 10:pp. 177{182.

Dolan, E. O. and More, J. J. (2002). Benchmarking optimization software with performance profiles. Mathematical programming, Vol. 19:pp. 201{213.

Fletcher, R. (1987). Practical method of Optimization. John Wiley, New York., 2nd edition.

Fletcher, R. and Reeves, C. M. (1964). Function minimization by conjugate gradients. Computer Journal., Vol. 7:pp. 149{154.

Gill, E. P., Murray, W., Saunders, A. M., and Wright, H. M. (1982). A note on a sufficient decrease criterion for a non-derivative steplength procedure. Jounal of Mathematical Programming, Vol. 23:pp. 349{352.

Goldstein, A. A. (1965). On steepest descent. J. Soc. Indust. Appl. Math. Ser. A Control, Vol. 3:pp. 147{161.

Hager, W. W. and Zhang, H. (2005). A new conjugate gradient method with guaranteed descent and an efficient line search. SIAM Journal on Optimization, Vol. 16:pp. 170{172.

Hestenes, M. R. and Stiefel, E. (1952). Methods of conjugate gradient for solving linear equations. J. Res. Nat. Bur. Stand., Vol. 49:pp. 409{436.

Jamil, M. and Yang, X. (2013). A literature survey of benchmark functions for global optimisation probllems. International Journal of Mathematical Modeling and Numerical Optimisation, Vol. 4, No. 2:pp. 150{194.

Jie, S. and Jiapu, Z. (2001). Global convergence of conjugate gradient methods without line search. Journal of Annals of Operation Research, Vol. 103:pp. 161{173. 20

Jinhong, H. and Genjiao, Z. (2013). A conjugate gradient method without line search and the convergence analysis. Fourth International Conference on Emerging Intelligent Data and Web Technologies., pages pp. 37{86.

Liu, Y. and Storey, C. (1992). E_cient generalized conjugate gradient algorithms. Journal of Optimization Theory and Application, Vol. 69:pp. 129{137.

Luenberger, D. G. (1984). Linear nonlinear programming. Addison-Wesley Publicizing Company.

Polak, E. and Ribiere, G. (1969). Note sur la convergence de directions conjugees. Rev. Francaise Informat Recherche Operationelle, Vol. 16:pp. 35{43.

Powell, M. J. D. (1976). Some global convergence properties of a variable metric for minimization without exact line search in nonlinear programming. SIAM-AMS Proceedings, IX, American Mathematical Society, Providence, RI, pages pp. 52{72.

Sun, W. and Yuan, Y. (2006). Optimization Theory and Methods. Springer.

Wolfe, P. (1969). Convergence conditions for ascent methods. SIAM Journal on optimization, pages pp. 226{235.

Wolfe, P. (1971). Convergence conditions for ascent methods. ii: Some corrections. SIAM Journal on optimization, Vol. 13, No. 2:pp. 185{900.

Wu, A. (2011). A nonlinear conjugate gradient method without line search. International Conference on Computation and Information Sciences, Vol. 4, No. 2:pp. 150{194.

Xiongda, C. and Jie, S. (2002). Global convergence of a two-parameter family of conjugate gradient methods without line search. Journal of Computational and Applied Mathematics, Vol. 146:pp. 37{45.

Zhang, B. (2010). Convergence of modified conjugate gradient methods without line search. Second International conference on Computational Intelligence and Natural Computing (CINC). 21

Zhang, J. Z., Deng, N. Y., and Chen, L. H. (1999). A new quasi-newton equation and related methods for unconstrained optimization. Journal of Optimization and Applications, Vol. 102:pp. 147{167.

Zhang, J. Z. and Xu, C. X. (2001). Properties and numerical performance of quasi-newton methods with modified quasi-newton equations. Journal of Computational and Applied Mathematics, Vol. 19:pp. 269{278.