IOSR Journal of Mathematics (IOSR-JM)
e-ISSN: 2278-5728, p-ISSN: 2319-765X. Volume 10, Issue 6 Ver. VI (Nov - Dec. 2014), PP 47-53
www.iosrjournals.org

A Diagnostic Treatment of Unconstrained Optimization Problems via a Modified Armijo line search technique.

Bamigbola O.M.[1], Adewumi A. O.[2], Adeleke O. J.*[3], Agarana M. C.[4]
[1]*Department of Mathematics, University of Ilorin, Ilorin. Nigeria*
[2]*School of Mathematics, Statistics and Computer Science, University of KwaZulu-Natal) Durban, South Africa.*
[*3]*(Corresponding Author)Department ofMathematics, Covenat University, Ota. Nigeria.*
[4]*Department of Mathematics, Covenat University, Ota. Nigeria..*

Abstract: *The study of optimization is getting broader with every passing day. Optimization is basically the art of getting the best results under a given set of circumstances. A very simple but important class of optimization problems is the unconstrained problems. Several techniques have been developed to handle unconstrained problems, one of which is the conjugate gradient method (CGM), which is iterative in nature. In this work, we applied the nonlinear CGM to solve unconstrained problems using inexact line search techniques. We employed the well-known Armijo line search criterion and its modified form.*

Keywords:*Line search, conjugate gradient method, unconstrained problems, step length.*

I. Introduction

A class of methods available to solve optimization problems is the descent methods. This method which involves the evaluation of the gradient of the objective function, are available to solve unconstrained minimization problems.This gradient method called the Conjugate Gradient Method (CGM) was proposed initially as many believed, in 1952 by Hestenes and Stiefel [6] for solving problems of linear algebra. Based on the concept of conjugacy, several CGMs have been developed by different researchers.
The conjugate gradient method (CGM) is used to solve the problem

Optimize $f(\underline{x}) = f(\underline{x}^{(0)}) + f'(\underline{x}^{(0)})^T(\underline{x} - (\underline{x}^{(0)}) + \frac{1}{2}(\underline{x} - \underline{x}^{(0)})^T f''(\underline{x}^{(0)})(\underline{x} - \underline{x}^{(0)})$ (1)

where $\underline{x}$, $\underline{x}^{(0)} \in \Re^N$, $f(\underline{x}^{(0)}) \in \Re$, $f'(\underline{x}^{(0)}) \in \Re^N$, and $f''(\underline{x}^{(0)}) \in \Re^N \times \Re^N$
with an iterative scheme:

$$\underline{x}^{(k+1)} = \underline{x}^{(k)} + \alpha_k \underline{d}^{(k)} \quad (2)$$

whereα_k is the step size and $\underline{d}^{(k)}$ is the direction of decent.

In order to obtain an$\alpha_k$for the iterative scheme indicated by (2) above, researchers have resorted to different methods. One of such methods is to find such an α^* which minimizes $f(\underline{x}^{(k)} + \alpha_k \underline{d}^{(k)})$.
i.e.,

$$\alpha^* = \operatorname{argmin} f(\underline{x}^{(k)} + \alpha_k \underline{d}^{(k)})$$

This closed form of obtaining α^* is known as exact line search.

Another method known as an inexact line search finds an approximate value of α_k numerically. This is achieved by employing various criteria which satisfy the descent property

$$\underline{g}^{(k)T} \underline{d}^{(k)} < 0 \quad (3)$$

II. Descent Methods for Unconstrained Optimization

In addition to function evaluation, the descent techniques require the evaluation of first and possibly higher order derivatives of the objective function. Due to this additional information regarding the objective function, these methods, also known as gradient methods, are generally more efficient than the direct search methods.

The gradient of a function, as defined by Rao [10], is the collection of partial derivative of a function with respect to each of its n variables. It is denoted by ∇f i.e.,

$$\nabla f(\underline{x}) = \begin{bmatrix} \frac{\partial f(\underline{x})}{\partial x_1} \\ \frac{\partial f(\underline{x})}{\partial x_2} \\ . \\ . \\ \frac{\partial f(\underline{x})}{\partial x_n} \end{bmatrix} \tag{4}$$

If there is a movement along the gradient direction from any point in n-dimensional space, the function value increases at the fastest rate. Thus, the gradient direction is called the direction of steepest ascent.

Since the gradient vector (4) represents the direction of steepest ascent, the negative of (4) denotes the direction of steepest descent. Thus, any method which makes use of the gradient vector can be expected to yield the minimum point faster than the one which does not make use of the gradient method.

III. Conjugate Gradient Method

The convergence rate of the steepest descent method described above can be greatly improved by modifying it into a conjugate gradient method (CGM). The CGM is a particular case of the conjugate direction method (CDM). One of the remarkable properties of CDM is its ability to generate, in a very economical fashion, a set of vectors $\underline{d}^{(i)}$, i = 1, 2, …,n having a property known as conjugacy, when applied to a quadratic functional. Therefore an n-dimensional quadratic function can be minimized in at most n-steps given n mutually conjugate directions.

The procedure used in the development of the CGM is analogous to the Gram-Schmidt orthogonalization procedure. The procedure sets up each new search direction as a linear combination of the previous search directions and the newly determined gradient.

i.e.

$$D = \begin{cases} -\underline{g}^{(i)}, & i = 0 \\ -\underline{g}^{(i)} + \beta_{i-1}\underline{d}^{(i-1)}, & i \geq 1 \end{cases} \tag{5}$$

where $\underline{g}^{(i)} = \nabla f(\underline{x}^{(i)})$ and β_{i-1} is a scalar known as the parameter of the method.

3.1Properties of Conjugate Gradient Method

The conjugate gradient method is a conjugate direction method with a very special property: in generating its set of conjugate vectors, it can compute a new vector $\underline{d}^{(k+1)}$ by using only the previous vector $\underline{d}^{(k-1)}$ of the conjugate set; $\underline{d}^{(k+1)}$ is automatically conjugate to these vectors.

Now, for a simple case of a quadratic objective function, the first search direction from the initial point $\underline{x}^{(0)}$ is in the direction of steepest descent; that is

$\underline{d}^{(0)} = -\underline{g}^{(0)}$

Thus

$$\underline{x}^{(i)} = \underline{x}^{(0)} + \alpha_0 \underline{d}^{(0)}$$

Where α_0 is the step length given by

$$\alpha_0 = \arg \min_{\alpha \geq 0} f(\underline{x}^{(0)} + \alpha \underline{d}^{(0)})$$

In a closed form, this can be show to give

$$\alpha_0 = \frac{\underline{g}^{(0)T}\underline{d}^{(0)}}{\underline{d}^{(0)T} H \underline{d}^{(0)}} \tag{6}$$

The general expression for eqn. (6) i.e

$$\alpha_k = \frac{\underline{g}^{(k)T}\underline{d}^{(k)}}{\underline{d}^{(k)}H\underline{d}^{(k)}} \tag{7}$$

can be shown as follows:
Let $\underline{d}^{(k)}$'s be linearly independent, then we can write

$$\underline{x}^* - \underline{x}^{(0)} = \alpha_0\underline{d}^{(0)} + \ldots + \alpha_{n-1}\underline{d}^{(n-1)} \tag{8}$$

for certain $\underline{d}^{(k)}$'s. Here $\underline{x}^*$ is the solution of $H\underline{x} = \underline{b}$
By multiplying the above by H and taking the scalar product with $\underline{d}^{(k)}$, we have

$$\alpha_k = \frac{\underline{d}^{(k)T}H(\underline{x} - \underline{x}^{(0)})}{\underline{d}^{(k)^T}H\underline{d}^{(k)}} \tag{9}$$

Using the iterative scheme

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

we have

$$x^{(1)} = \underline{x}^{(0)} + \alpha_0\underline{d}^{(0)}$$
$$x^{(2)} = \underline{x}^{(1)} + \alpha_1\underline{d}^{(1)} = \underline{x}^{(0)} + \alpha_1\underline{d}^{(1)}$$
$$.$$
$$.$$
$$x^{(k)} = \underline{x}^{(0)} + \alpha_0\underline{d}^{(0)} + \ldots + \alpha_{k-1}\underline{d}^{(k-1)}$$

Therefore,

$$\underline{x}^{(k)} = \underline{x}^{(0)} + \alpha_0\underline{d}^{(0)} + \ldots + \alpha_{k-1}\underline{d}^{(k-1)}$$

By multiplication with H and taking the scalar product with $\underline{d}^{(k)}$ we obtain (from H –orthogonality of $\underline{d}^{(k)}$'s)

$$d^{(k)T}H(\underline{x}^{(k)} - \underline{x}^{(0)}) = 0$$

From (6), we can write α_k as

$$\alpha_k = \frac{\underline{d}^{(k)T}H(\underline{x}^* - \underline{x}^{(k)} + \underline{x}^{(k)} - \underline{x}^{(0)})}{\underline{d}^{(k)^T}H\underline{d}^{(k)}}$$

$$= \frac{\underline{d}^{(k)T}H(\underline{x}^* - \underline{x}^{(k)}) + \underline{d}^{(k)^T}H(\underline{x}^{(k)} - \underline{x}^{(0)})}{\underline{d}^{(k)^T}H\underline{d}^{(k)}}$$

Therefore,

$$\alpha_k = \frac{\underline{d}^{(k)T}H(\underline{x}^* - \underline{x}^{(k)})}{\underline{d}^{(k)^T}H\underline{d}^{(k)}}$$

$$= \frac{\underline{d}^{(k)T}(H(\underline{x}^* - H\underline{x}^{(k)})}{\underline{d}^{(k)^T}H\underline{d}^{(k)}}$$

$$= \frac{\underline{d}^{(k)^T}(\underline{b} - H\underline{x}^{(k)})}{\underline{d}^{(k)^T}H\underline{d}^{(k)}}$$

Since $\underline{g}^{(k)} = H\underline{x}^{(k)} - \underline{b}$, we get

$$\alpha_k = \frac{\underline{g}^{(k)^T}\underline{d}^{(k)}}{\underline{d}^{(k)^T}H\underline{d}^{(k)}}$$

In the next stage, $\underline{d}^{(k+1)}$ is taken as a linear combination of $\underline{g}^{(k+1)}$ and $\underline{d}^{(k)}$
i.e.

$$\underline{d}^{(k+1)} = \underline{g}^{(k+1)} + \beta_k\underline{d}^{(k)},\ k = 0.\ 1,\ 2, \tag{10}$$

The coefficients β_k, k = 0, 1, 2… known as conjugate gradient parameters are chosen in a such a way that $\underline{d}^{(k+1)}$ and $d^{(k)}$ are conjugate with respect to H. By premultiplyng (7) by $d^{(k)T}H$ and imposing the condition

$$\underline{d}^{(k)T} H \ \underline{d}^{(k+1)} = 0,$$

we obtain

$$\beta_k = \frac{\underline{g}^{(k+1)T} H \underline{d}^{(k)}}{\underline{d}^{(k)T} H \underline{d}^{(k)}} \qquad (11)$$

The above discussion completely describes the stages involved in the conjugate gradient algorithm for quadratic functional which we now outline below.

IV. Conjugate Gradient Algorithm for Nonquadratic Functions

In the previous sections, we have shown that the conjugate gradient algorithm is a conjugate direction method, and therefore minimizes a positive definite quadratic function on n variable in n steps. The conjugate gradient algorithm for a quadratic function can be extended to general nonlinear functions by interpreting $f(\underline{x}) = \frac{1}{2}\underline{x}^{T+}H\underline{x} + \underline{x}^T\underline{b} + c$ as a second-order Taylor series approximation of the objective function. Near the solution such functions behave approximately as quadratics, as suggested by the Taylor series expansion. For quadratic, the Hessian matrix, H, is constant. However, for a general nonlinear function the Hessian is a matrix that has to be reevaluated at each iteration of the algorithm. This can be computationally very expensive. Hence, an efficient implementation of the conjugate gradient algorithm that eliminates the Hessian evaluation at each step is desirable.

To derive the nonlinear conjugate gradient method, there are generally two changes to the conjugate gradient algorithm: it becomes more complicated to compute the step length α, and there are several different choices for β_k, the conjugate gradient parameter
Because

$$\alpha_k = \arg\min_{\alpha \geq 0} \ f(\underline{x}^{(k)} + \alpha \, \underline{d}^{(k)}), \qquad (12)$$

the close-form or the exact formula for α_k in the CG algorithm can be replaced by a numerical line search procedure. The idea is as in the CG for quadratic function whereby a value of α_k that minimizes $f(\underline{x}^{(k)} + \alpha\underline{d}^{(k)})$ is found by ensuring that the gradient is orthogonal to the search direction. Since we are dealing with nonquadratic function f, and α_k is determined from a one-dimensional optimization that minimizes $f(\underline{x}^{(k)} + \alpha\underline{d}^{(k)})$, we can do this numerically using line search techniques. The only consideration will then be how the line search method with $\underline{d}^{(k)}$ can be applied to the minimization of nonquadratic functions. In the CGA for quadratic function, there are several equivalent expressions for the value of β_k. In nonquadratic CG, these different expressions are no longer equivalent.

Since it is possible to eliminate H from (11), the resulting algorithms now only depend on function and gradient values at each iteration.By interpreting (5) as the second-order Taylor series approximation of the non-quadratic objective function with H the approximation of the Hessian and β_k determined from H, we proceed as follows:

$\beta_k = \frac{\underline{g}^{(k+1)T} H \underline{d}^{(k)}}{\underline{d}^{(k)^T} H \underline{d}^{(k)}}$ and $\alpha_k H\underline{d}^{(k)} = \underline{g}^{(k+1)} - \underline{g}^{(k)}$ for the quadratic function, we replace the calculation for β_k by

$$\beta_k = \frac{\underline{g}^{(k+1)T}(\underline{g}^{(k+1)} - \underline{g}^{(k)})}{\underline{d}^{(k)^T}(\underline{g}^{(k+1)} - \underline{g}^{(k)})} \qquad (13)$$

With several researchers working extensively on the modification of eqn (13), several formula for β_k has emerged corresponding to different CGMs. Andrei [1] has identified as many as forty (40) CGMs and yet the number of CGMs available now is on the increase. The most recent of the CGMs is that of Bamigbola et al. [2].

The following are few of the β_k s available in literature.

1. Hestenes – Stiefel (1952): $\beta_k^{Hs} = \frac{\underline{g}^{(k+1)T}\underline{y}_k}{\underline{d}^{(k)^T}\underline{y}_k}$ [5]

2. Polak-Ribiere-Polyak (1969): $\beta_k^{PRP} = \dfrac{\underline{g}^{(k+1)T}\underline{y}_k}{\underline{g}^{(k)T}\underline{g}_k}$ [8, 9]

3. Fletcher-Reeves (1964): $\beta_k^{FR} = \dfrac{\underline{g}^{(k+1)T}\underline{g}^{(k+1)}}{\underline{g}^{(k)T}\underline{g}^{(k)}}$ [4]

4. Dai-Yuan (2000): $\beta_k^{DY} = \dfrac{\underline{g}^{(k+1)T}\underline{g}^{(k+1)}}{\underline{g}^{(k)T}\underline{y}_k}$ [3]

5. Liu-Storey (1992): $\beta_k^{LS} = \dfrac{\underline{g}^{(k+1)T}\underline{y}_k}{\underline{g}^{(k)T}\underline{d}^{(k)}}$ [7]

6. Conjugate Descent (1997): $\beta_k^{CD} = \dfrac{-\underline{g}^{(k+1)T}\underline{g}^{(k+1)}}{\underline{g}^{(k)T}\underline{d}^{(k)}}$ [5]

7. Bamigbola-Ali-Nwaeze (2010): $\beta_k^{BNA} = \dfrac{-\underline{g}^{(k+1)T}\underline{y}_k}{\underline{g}^{(k)T}\underline{y}_k}$ [2]

where $\underline{y}_k = \underline{g}^{(k+1)} - \underline{g}^{(k)}$

The corresponding methods for the above β_ks are here denoted respectively as HS, PRP, FR, DY, LS, CD and BAN.

V. Armijo Line Search Criterion

Among the simplest line search algorithms is Armijo's method. This method finds a step size $\bar{\alpha}$ that satisfies a condition of sufficient decrease of the objective function and implicitly a condition of sufficient displacement from the current point $\underline{x}^{(k)}$.

This Armijo's condition is given as

$$\phi(\alpha) \leq \phi(0) + \alpha\rho\phi'(0)$$

In many cases it may be necessary to impose stronger conditions on the step size α.

5.1Armijo Line Search

Step 1: Choose $\rho\in(0, ½)$; set$\alpha = 1$ and $\varepsilon = 0.5$

Step 2: While $f(\underline{x}^{(k)} + \alpha_k \underline{d}^{(k)}) > f(\underline{x}^{(k)}) + \rho\alpha\nabla f(\underline{x}^{(k)})^T \underline{d}^{(k)}$, take $\alpha = \varepsilon\alpha$

Step 3: Terminate with $\alpha_k = \alpha$, set $\underline{x}^{(k+1)} = \underline{x}^{(k)} + \alpha_k \underline{d}^{(k)}$

5.2Modified Armijo Line Search

Our modification of the above Armijo line search algorithm is what follows. The basic ideal was to specify an interval for the rate, ε, at which the step size, α, reduces.

Step 1: Choose $\rho\in(0, ½)\varepsilon\in(0, 1)$ set $\alpha = 1$

Step 2: While $f(\underline{x}^{(k)} + \alpha_k \underline{d}^{(k)}) > f(\underline{x}^{(k)}) + \rho\alpha\nabla f(\underline{x}^{(k)})^T \underline{d}^{(k)}$, take $\alpha = \varepsilon\alpha$ for some $\varepsilon\in(\varepsilon_1, \varepsilon_2) = (0.1, 0.5)$ i.e.,ε is chosen a new randomly from the open set (0.1, 0.5).

Step 3: Terminate with $\alpha_k = \alpha$, set $\underline{x}^{(k+1)} = \underline{x}^{(k)} + \alpha_k \underline{d}^{(k)}$

VI. Computational Details

The computational experiments carried out in this research incorporated the inexact Algorithms in section 5 into the nonlinear CGM Algorithm discussed above. Our aim is to perform experiments that aid in measuring the effectiveness of two of theCGMs i.e., the Fletcher-Reeves and Polak-Ribiere-Polyakand the inexact algorithms in the last section. To achieve this, ten (10) nonlinear unconstrained optimization test functions by Andrei [1] were used for computational illustrations. These functions range from polynomial functions to nonpolynomial objective functions. Our computational instrument was a Matlab Code designed for the purpose of this research.

VII. Results

The tables below present the outcome of the computational experiments on the test functions. The following notations were used; f* – optimal value of the objective function, ||g*|| – the norm of the gradient of f*, n – dimension, CE – computational example, Exec – program execution time, and Iter – number of iterations.

Table 1: Numerical Result using Armijo Line Search

CE	FR				PRP			
	Iter	**f ***	**||g*||**	**Exec**	**Iter**	**f ***	**||g*||**	**Exec**
Extended Rosenbrock Funtion	107	8.86e –15	3.5e – 0.7	1.19	174	1.79e –13	9.0e – 07	3.73
	520	3.07e– 13	7.9e – 07	11.58	178	4.14e– 15	5.9e – 07	6.48
Diagonal 4 Function	30	1.43e – 16	3.5e – 07	0.46	37	2.35e – 16	4.0e – 07	0.92
	27	4.00e – 16	5.6e – 07	0.72	37	4.71e – 16	5.6e – 07	1.43
Extended Himmelblau Function	34	3.9e– 15	7.2e – 07	0.86	34	3.5e– 15	6.2e – 07	0.73
	36	1.3e – 15	3.4e – 07	1.38	34	7.0e – 15	8.8e – 07	1.28
Extended Beale Function	102	1.8e – 13	5.5e – 07	3.30	221	8.5e – 15	9.0e – 07	16.37
	178	3.5e – 14	5.8e – 07	12.09	220	3.7e – 13	7.6e – 07	33.40
Modified Extended Beale Function	195	1.5e02	8.0e – 07	8.76	234	1.5e02	1.2e – 06	13.52
	806	3.1e02	1.4e – 06	79.81	240	3.1e02	1.4e – 06	26.53
Extended Block Diagonal BD1 Function	93	2.8e – 09	4.6e – 07	1.64	3	6.1e – 60	3.1e – 45	0.29
	71	1.2e – 12	8.4e – 08	2.78	3	1.2e – 59	4.4e – 45	0.55
Generalized Tridiagonal 1 Function	189	6.4e –02	3.8e – 01	3.52	36	6.1e –15	2.2e – 07	1.21
	97	5.2e– 14	6.8e – 07	2.65	36	9.7e– 17	2.8e – 08	2.37
Generalized White Holst Function	82	4.0e00	5.5e – 07	1.78	113	4.0e00	6.2e – 07	3.63
	82	4.0e00	5.5e – 07	3.49	113	4.0e00	6.2e – 07	7.98
Extended Tridiagonal 1 Function	481	5.0e03	3.4e – 05	10.37	563	5.0e03	7.9e – 07	11.29
	162	1.0e04	1.3e – 05	6.76	626	1.0e04	1.2e – 06	33.44
Extended Freudenstein and Roth Function	211	1.2e05	8.4e – 06	5.76	453	2.2e – 10	5.8e – 05	22.93
	181	2.4e05	2.2e – 05	9.98	274	2.2e – 06	6.5e – 03	30.69

Table 2: Numerical Results Using Modified Armijo Line Search.

CE	FR				PRP			
	Iter	***f ****	**‖g*‖**	**Exec**	**Iter**	***f****	**‖g*‖**	**Exec**
Extended Rosenbrock Funtion	107	8.7e-15	3.5e-7	1.25	174	1.8e-13	9.0e-7	3.74
	520	3.1e-13	7.9e-7	11.59	178	4.1e-15	5.9e-7	6.94
Diagonal 4 Function	25	1.2e-15	9.7e-7	0.56	37	2.3e-16	4.0e-7	0.90
	27	4.0e-16	5.6e-7	0.88	37	4.7e-16	5.6e-7	1.59
Extended Himmelblau Function	53	6.0e-15	5.7e-7	0.83	34	3.5e-15	6.2e-7	0.81
	53	1.2e-14	8.2e-7	1.48	34	7.0e-15	8.8e-7	1.25
Extended Beale Function	102	1.8e-13	5.5e-7	3.37	221	8.4e-15	9.0e-7	16.38
	178	3.5e-14	5.8e-7	11.98	220	3.7e-13	7.6e-7	33.86
Modified Extended Beale Function	195	1.5e02	8.0e-7	4.80	234	1.5e02	1.2e-6	6.85
	909	3.1e02	2.8e-7	50.88	240	3.1e02	1.4e-6	13.63
Extended Block Diagonal BD1 Function	93	2.8e-9	1.6e-7	1.00	3	6.1e-60	3.1e-45	0.32
	71	1.2e-12	8.4e-8	1.48	3	1.2e-59	4.4e-45	0.42
Generalized Tridiagonal 1 Function	189	6.4e-02	3.8e-1	1.85	36	6.1e-15	2.2e-7	0.78
	97	5.2e-14	6.8e-7	1.55	36	9.7e-17	2.8e-8	1.25
Generalized White Holst Function	82	4.0e00	5.5e-7	1.04	113	4.0e00	6.2e-7	2.00
	82	4.0e00	5.5e-7	1.99	113	4.0e00	6.2e-7	4.15
Extended Tridiagonal 1 Function	481	5.0e03	3.4e-5	5.35	563	5.0e03	7.9e-7	5.65
	162	1.0e04	1.3e-5	2.96	626	1.0e04	1.2e-6	12.99
Extended Freudenstein and Roth Function	211	1.2e05	8.4e-6	3.17	591	4.2e-14	8.11e-7	14.69
	181	2.4e05	2.2e-5	5.00	514	1.2e-14	8.4e-7	26.43

VIII. Discussion of Result

From the above tables, it is clear that our modification of the Armijo line search criterion is still very much in place. At a specified tolerance level of ‖**g***‖ = 1.0e-6, thoughwe obtained identical results but the rate of convergence was moderately improved in the modified algorithm.

IX. Conclusion

We conclude by affirming that the modified algorithm in this work was effective in the computational treatment of unconstrained optimization problems.

References

[1]. Andrei, N. (2004). Unconstrained optimization test functions. Unpublished manuscript; Research Institute for Informatics. Bucharest 1, Romania.

[2]. Bamiigbola, O.M., Ali, M. And Nwaeze, E. (2010). An efficient and convergent method for unconstrained nonlinear optimization. Proceedings of International Congress of Matthematicians. Hyderabad, India.

[3]. Dai, Y and Yuan, Y. (2000). A nonlinear conjugate gradient with a strong global convergence properties: SIAM Journal on Optimization. Vol. 10, pp. 177-182.

[4]. Fletcher, R. and Reeves, C.M. (1964). Function minimization by conjugate gradients. Computer Journal. Vol 7, No. 2.

[5]. Fletcher, R. (1997). Practical method of optimization. Second ed. John Wiley, New York.

[6]. Hestenes, M.R. and Stiefel, E. (1952). Method of conjugate gradient for solving linear equations. J. Res. Nat. Bur. Stand. 49.

[7]. Liu, Y. and Storey, C. (1992). Efficient generalized conjugate gradient algorithms. Journal of Optimization Theory and Application. Vol. 69, pp. 129-137.

[8]. Polak, E. and Ribiere, G. (1969). Note sur la convergence de directions conjugees. Rev. FrancaiseInformatRechercheOperationelle. 3e Annee 16, pp. 35-43.

[9]. Polyak, B.T. (1969). The conjugate gradient in extreme problems. USSR Comp.Math. Math. Phys. 94-112.

[10]. Rao, S.S. (1980). Optimization theory and application. Second edition, Wiley Eastern Ltd., New Delhi.