A new self scaling PCG algorithm with dynamical retards

ABBAS Y. AL-BAYATI & ASEEL MOAYAD QASIM
Department of mathematics
College Of Computer Sciences and Mathematics
Mosul University

Received **Accepted**
04/03/2007 **09/05/2007**

الملخص

ان مسالة ايجاد الحل الامثل لنظام معين كدالة للزمن يمكن صياغتها بشكل دالة تعظيم او تصغير لدالة الكلفة غير الخطية. وقد وجد بشكل خاص أن خوارزميات البرمجة الديناميكية كفوءة لحل هذه المسائل مقارنة مع محاولات طرق البرمجة التكرارية غير الخطية.

وفي عملنا هذا , عممت طرق الانحدار المتدرج مع الاعاقة الديناميكية ورفقت بطريقة ديناميكية مع الاستراتيجيات غير الرتيبة للحصول على نوع جديد من خوارزميات التصغيرللدوال غيرالتربيعية التي تتعامل بكفاءة مع المسائل اللاخطية في الامثلية ذات القياس العالي.

Abstract

The problem of optimizing a certain systems as a function of time may be formulated in terms of maximizing or minimizing a non-linear cost function. It was found that a Dynamic Programming (DP) algorithm was particularly efficient procedure for solving this problem compared with an alternative nonlinear programming method.

In this work, a new CG method with dynamical retards is generalized and combined in a dynamical way with non-monotone globalization strategies to obtain a new type CG-algorithm for minimizing non-quadratic functions that can deal efficiently with large scale nonlinear optimization problems.

1. Introduction:

Conjugate gradient (CG) algorithms have advantage over both modified–Newton and variable metric (VM) algorithms for unconstrained optimization of not requiring a matrix store. They tend, however, to require more function evaluations to obtain the minimum than the latter algorithms and therefore their use is usually restricted to the solution of problem in a large number of dimensions, or to computers with a restricted store.

We consider the unconstrained minimization problem

$$\min_{x \in R^n} f(x), \qquad \dots (1)$$

where $f:\Re^n \longrightarrow \Re$ is continuously differentiable and its gradient is available. We are interested in the large–scale case for which the Hessian of f is either not available or requires a prohibitive amount of storage and computational cost.

The most important features of these methods are that only gradient directions are used, that the memory requirement are minimal, and they do not involve a decrease in the objective function, which allows fast local convergence.

In this paper, we extended the gradient method and introduce a new gradient method with dynamical retards to find the unique global minimizer of the quadratic function of form

$$f(x)=\frac{1}{2}x^T G x - b^T x \ , \qquad \ldots(2)$$

where $G \in \Re^{n\times n}$ is large, sparse, symmetric and positive definite.

For the non-quadratic case, the method needs to be incorporated in a globalization strategy, since the method does not enforce decrease in the objective function, a non-monotone line search strategy will be used.

In particular, the non-monotone line search technique introduced by Grippo, Lampariello and lucidi [8] has proved to be very effective for large scale optimization problems. This line search essentially enforces the following condition

$$f(X_{k+1}) \le \max_{0\le j\le M} f(X_{k-j}) + \gamma g_k^T (X_{k+1} - X_k), \qquad \ldots (3)$$

Where M is a non-negative integer and γ is a small positive number. This condition leads us to create two algorithms; the first which is called (The original) using this condition only, and the second which called (The new) using Armijo rules with BFGS updating formula and self scaling Variable Metric (VM) update for the scalar Hestense and Stiefel (HS) [9].

This paper is organized as follows: In the next section we discuss the line search procedure and then talking about Steepest Descent (SD) method and in the third section we describe a review of conjugate gradient method and in the fourth section the non-monotone gradient methods will be describe and in the fifth section we describe VM methods and the new Preconditional Conjugate Gradient (PCG) algorithm and some new theorems, while section six gives the numerical results and the conclusions, sections seven and eight contain the Appendix and the References.

2. Line search procedure:

One-dimensional search is the backbone of many algorithms for solving a nonlinear programming algorithms precede as follows: given a point x_k, find a direction vector d_k and then a suitable step-size λ_k, yielding a new point $x_{k+1} = x_k + \lambda_k d_k$; the process is then repeated. Finding the step-size λ_k involves solving the sub problem to minimize $f(x_k + \lambda d_k)$, which is a one-dimensional search problem in the variable λ.
The minimization may be over all real λ, non-negative λ ,or such that $x_k + \lambda d_k$ is feasible. Consider a function θ of one variable λ to be minimized. One approach to minimizing θ is to set the derivative θ' equal to 0 and then solve for λ. Note, however, that θ is usually defined implicitly in terms of a function f of variables. In practice, given the vector x and d, $\theta(\lambda)=f(x+\lambda d)$. If f is not differentiable , then θ will not be differentiable. if f is differentiable, then $\theta'(\lambda) = d'\nabla f(x+\lambda d) = 0$. Therefore, to find a point λ with $\theta'(\lambda)=0$, we have to solve the equation $d'\nabla f(x+\lambda d) = 0$, which is usually nonlinear in λ. Furthermore, λ satisfying $\theta'(\lambda) = 0$ is not necessarily a minimum; it may be a local minimum , a local maximum , or even a saddle point , for these reasons , and except for some special cases , we avoid minimizing θ by letting its derivative be equal to zero. Instead, we resort to numerical techniques for minimizing the function θ. In our work we use this technique to find λ as follows:

find z = min (k ,M) , where k is the count , M is non negative integer and for $0 \le j \le z$, find Max $f_{(k-j)} - \gamma.\lambda.g_k g_k^T$ where g is the gradient of the function f , $\gamma \in (0,1)$, $\delta \in (0,1)$. If $f(x_{k+1}) > \max f(k-j) - \lambda.\gamma.g_k.g_k^T$ choose $\lambda = \delta.\lambda$, $x_{k+1} = x_k + \lambda d_k$ else $\lambda_k = \lambda$, $x_{k+1} = x_k + \lambda_k . d_k$ update λ by take $\alpha = \frac{v^T y}{v^T v}$ and $\lambda = 1/\alpha$.

2.1 Steepest -Descent (SD):
The Steepest Descent (SD) method is particularly useful when the dimension of the problem is very large, however, it may generate short zigzagging displacements in a neighborhood of a solution.
For simplicity, we denote $\nabla f(x_k)$ by g_k , $f(x_k)$ by f_k and $f(x^*)$ by f^*, respectively, where x^* denotes a local minimizer of *f*.
In the algorithmic framework of (SD) method the iterative formula
$x_{k+1} = x_k + \lambda_k d_k, k = 0,1,2,.$ ... (4)
Where d_k satisfies the relation

$g_k^t d_k < 0$, … (5)

Which guarantees that d_k is a descent direction of f(x) at x_k. In order to guarantee the global convergence, it is usually required the descent condition

$g_k^T d_k \le -c \|g_k\|^2$, ... (6)

Where c > 0 is a constant . The angle property

$\cos(g_k^T d_k) = -\frac{g_k^T d_k}{\|g_k\| . \|d_k\|} \ge \eta_0$, ... (7)

is often used in many situations, with $\eta_0 \in (0,1]$.

Observe that, if $\|g_k\| \neq 0$ then $d_k = -g_k$ satisfies (5) , (6) and (7) simultaneously.

There are many alternative line-search rules to choose λ_k along the ray $d_{k+1} = \{x_k + \lambda_k d_k \mid \lambda > 0\}$.namely:

(a) Minimization Rule. At each iteration, λ_k is selected so that

$f(x_k + \lambda_k d_k) = \min_{\lambda > 0} f(x_k + \lambda_k d_k)$ … (8)

(b) Armijo Rule. Set scalars d_k, γ, L and δ with $d_k = -\frac{g_k^T d_k}{L\|d_k\|^2}$, $\gamma \in (0,1)$,

$L > 0$ And $\delta \in (0,1/2)$.Let λ_k be the largest λ in $\{d_k, \gamma d_k, \gamma^2 d_k, ...\}$ such that

$f_k - f(x_k + \lambda_k d) \ge -\delta \lambda g_k^T d_k$ ... (9)

(c) Approximate Minimization Rule. At each iteration, λ_k is selected so that

$\lambda_k = \min\{\lambda \mid g(x_k + \lambda d)^T d_k = 0, \lambda > 0\}$ ...(10)

For more details see [13].

In our work we choose Armijo rule to describe the new proposed algorithm in section (5.5).

3. Review of conjugate gradient method:

Conjugate gradient (CG) methods were first used to solve the general unconstrained problem by Fletcher and Reeves [6]. Their algorithm (or simple variants) is still frequently used, especially for problems with a large number of variables since they require only a few vectors of length n to be stored.

Given a symmetric positive definite matrix G, the finite set of non-null vectors $\{d_1, d_2, ..., d_k\}$ is said to form a conjugate set if

$d_i^T G d_j = 0$ for all i ≠ j …(11)

The CG-method, which we shall now derive, is based on a successive construction of conjugate search directions. Suppose for now that we know the matrix G defining the quadratic function.

$q(x) = \frac{1}{2}(x^T G x) + b^T x$; ...(12)

then we can construct a set of conjugate directions $\{d_1, d_2, ..., d_n\}$ from an arbitrary set of linearly independent direction $\{r_1, r_2, ..., r_n\}$ by a Gram - Schmidt process in the following way [3] :

Let $d_1 = r_1$, For i=1,2,……n determine successively

$d_i = r_i + \sum_{j=1}^{i-1} c_{ij} d_j$, ... (13)

where the coefficient c_{ij} must be chosen so that

$d_i^T G d_k = 0$ for k=1,2,……,i-1 ….(14)

From (13) we get $d_i^T G d_k = r_i^T G d_k + \sum_{j=1}^{i-1} c_{ij} d_j^T G d_k$ … (15)

By the obvious inductive assumption, substitution of (14) into (15) yields:

$c_{ij} = [-r_i^T G d_j \,/\, d_j^T G d_j]$ For j=1, 2,…, i-1 …(16)

now $d_j \neq 0$, since r_j are assumed to be linearly independent, hence

$d_j^T G d_j > 0$ and $c_{ij} = 0$ is well–defined.

Suppose we want to minimize the quadratic function q without first evaluating the Hessian G, but suppose also that we can compute the gradient g since

$x_{k+1} = x_k + \lambda_k d_k$ ….(17a)

$g_{k+1} = G x_{k+1} + b$, …(17b)

$y_k = g_{k+1} - g_k = \lambda_k G d_k$, ….(17c)

$v_k = x_{k+1} - x_k$, … .(17d)

Let the set $\{r_1, r_2, ..., r_n\}$ be chosen as the set $\{d_1, -g_2, ..., -g_n\}$ where d_1 is an arbitrary downhill direction and $\{g_2, g_3, ..., g_n\}$ are determined successively. Substituting now (17) and using $r_i = -g_i$ for i ≥2 into (16) we obtain

$$c_{ij} = [g_i^T (g_{j+1} - g_j) \big/ d_j^T (g_{j+1} - g_j)] = [g_i^T y_j \big/ d_j^T y_j], \ldots..(18)$$

for q with an exact line search (ELS) we have

$d_i^T g_{j+1} = 0$ for i = 1, 2 ,3 , j , …(19)

also since $\{d_{2,} d_3, \ldots d_j\}$ were constructed as a linear combination of $\{g_{2,} g_3, \ldots g_j\}$ we see that

$g_i^T g_j = 0$ for $2 \leq$ i $<$ j …..(20)

this orthogonally relation makes $c_{ij} = 0$ for j = 2, 3, i-2

……(21)

Finally, (13) reduces to the following:

$d_2 = -g_2 + [g_2^T y_1 \big/ d_1^T y_1] d_1,$ …(22a)

$d_{k+1} = -g_{k+1} + \beta_k d_k + \gamma_k d_1,$ …(22b)

$\beta_k = [g_{k+1}^T y_k \big/ d_k^T y_k],$ …(22c)

$\gamma_k = [g_{k+1}^T y_1 \big/ d_1^T y_1],$ … .(22d)

This is the Beale CG-method [3] in regular CG-methods the first direction d_1 is not arbitrary, but it is taken as $-g_1$.

Then in this case

$g_i^T g_j = 0$ For $1 \leq$ i $<$ j ….(23)

3.1 Regular CG-method:

The regular CG-method can be defined as follows:

given an arbitrary starting point x_i ,define the first search direction by $d_1 = -g_1$ and for k = 1,2,3,…. iterate with

$x_{k+1} = x_k + \lambda_k d_k,$ ….(24a)

$\beta_k = [g_{k+1}^T y_k \big/ d_k^T y_k]$, ….(24b)

$d_{k+1} = -g_{k+1} + \beta_k d_k,$ ….(24c)

the formula (24b) is called the Hestenes and Stiefel [9] form of β_k .It can also be written as:

$$\beta_k = \left[g_{k+1}^T y_k \Big/ g_k^T g_k \right], \qquad \ldots(25a)$$

the Polak-Ribiere [11] form ;or

$$\beta_k = \left[g_{k+1}^T y_{k+1} \Big/ g_k^T g_k \right], \qquad \ldots.(25b)$$

the Fletcher-Reeves [7] form; or

$$\beta_k = \left[-g_{k+1}^T g_{k+1} \Big/ g_k^T d_k \right], \qquad \ldots (25c)$$

Dixon [5] form.

For the quadratic function in (2) with ELS these four formulae for β_k are equivalent, but for general function they are not the same and give different algorithms, however, for any arbitrary function f, the CG-method (24) generates downhill directions provided ELS are used since

$$d_{k+1}^T g_{k+1} = \left(-g_{k+1} + \beta_k d_k \right)^T g_{k+1}, \qquad \ldots(26)$$

$$= \left(-g_{k+1}^T g_{k+1} \right) \prec 0, \qquad \ldots (27)$$

In fact the ELS required can be relaxed since it is clear from (27) that a downhill direction can be obtained whenever

$$\beta_k d_k^T g_{k+1} < g_{k+1}^T g_{k+1}, \qquad \ldots(28)$$

In practice this is usually replaces by the slightly stronger condition:

$$\beta_k d_k^T g_{k+1} < \rho (g_{k+1}^T g_{k+1}), \qquad \ldots(29)$$

Where ρ is some small constant, say 0.2 or 0.1 (very small positive number) and this condition is attainable in practice for any continuous function.

4. Non-Monotone Gradient Methods:

For the minimization of non-quadratic functions ,non-monotone methods like the gradient method ,need to be incorporated with a globalization strategy .The main idea is to accept the step if it is satisfies a weak condition of the form given by (3) ,when M > 0 ; this condition allows the objective function to increase at some iterations and still guarantees global convergence first in this section we would like to present a general non-monotone gradient algorithm for which we can establish classical convergence results.

4.1 Non-monotone line search algorithm:

We adopted a non-monotone line-search strategy, we do not impose decrease of the objective function at every iteration . Instead, we choose a positive integer M at the beginning of the process and we accept a trial point when a sufficient decrease is obtained in relation to the maximum functional value among the M+1 last iterations.

assume that $\lambda \in (0,1)$ is given independently of the iteration number k and that d_k has been computed .

Step 1 : set $\lambda \longleftarrow 1$.

Step 2 :set $x_{k+1} = x_k + \lambda d_k$.

Step 3: if $f(x_k + \lambda_k d_k) \le \underset{0 \le j \le \min(k,M)}{\text{Max}} f(x_{k-j}) + \gamma g_k^T (x_{k+1} - x_k)$

then define $\lambda_k = \lambda$ and finish the line search.

If the condition in step 3 dose not hold, define $\lambda_{new} \in [0.1\lambda, 0.9\lambda]$

Set $\lambda \longleftarrow \lambda_{new}$ and goto step 2

for more detail see [10].

4.2 Original Algorithm:

Outlines of the original non-monotone algorithm [12]:

Step1: Given x_0, $\varepsilon = 0.05$, n is the dimension of the problem , Ac=1*E-4

k =0, integer M > 0 , $\gamma \in (0,1)$, $0 < \delta < 1$, $\alpha = \max(1, \min(1/\varepsilon, \|g\|)$,

$\lambda = 1/\alpha$

Step2: $d_k = -g_k$

Step 3: If $\|g_{k+1}\| < Ac$ stop else continue.

Step4: $x_{k+1} = x_k + \lambda d_k$

Step5: Find z=min (k ,M),$0 \le j \le z$, Max$f_{(k-j)} - \gamma.\lambda.g_k^T.g_k$

If $f(x_{k+1}) > \max f(k-j) - \lambda.\gamma.g_k.g_k^T$ choose $\lambda = \delta.\lambda$,

$x_{k+1} = x_k + \lambda .d_k$

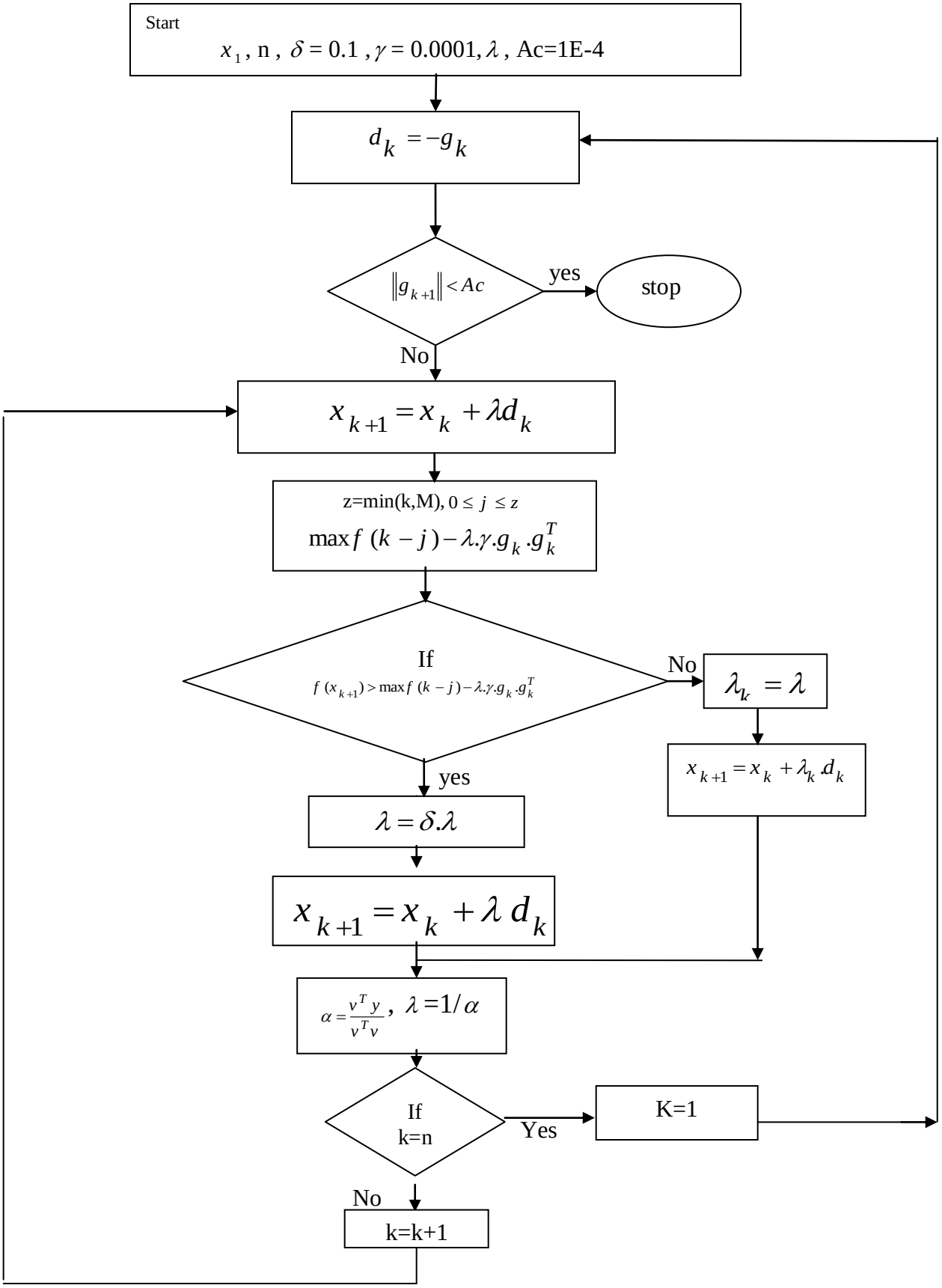
else $\lambda_k = \lambda$, $x_{k+1} = x_k + \lambda_k . d_k$, $\alpha = \frac{v^T y}{v^T v}$, $\lambda = 1/\alpha$

Step 6: $g_{k+1} = \nabla f(x_{k+1})$, If k = n Consider k = 1 and go to (step 2), otherwise k = k+1 and go to (step 4)

Step7: End

In the next section we are going to draw the flow chart of the original non-monotone algorithm as follows:

4.3 Flow chart of the original non-monotone algorithm:



4.4 Theorem:

Assume that $\Omega=\{x\in\Re^n : f(x)\le f(x_0)\}$ is abounded set. If f is continuously differentiable on an open set that contains Ω then algorithm 4.2 is well defined and any accumulation point of the sequence $\{x_k\}$ that generates stationary point.

For the details of the proof see [12].

5. Variable Metric (VM) methods:

Quasi-Newton methods are probably the most popular general purpose algorithm for unconstrained optimization problems .Many QN-method are advantageous due to their fast convergence and absence of second order derivatives computation.

For the QN-methods assume that at the k- th iteration at approximation point x and n×n matrix H_k are available, then the methods proceed by generating a sequence of approximation points via the equation:

$x_{k+1}=x_k+\lambda_k d_k \quad , d_k=-H_k g_k$

where H_k is an approximation of G_k^{-1}which is corrected for updating from iteration to iteration .In general H_k is symmetric and positive definite ,there are different choices of H_k see [7] we list here some most popular forms

$$H_{k+1}^{SR1}=H_k+\frac{(v_k-H_k y_k)(v_k-H_k y_k)^T}{(v_k-H_k y_k)^T y_k}, \qquad \dots (30)$$

is called rank one correction formulae, where

$v_k=x_{k+1}-x_k$ and $y_k=g_{k+1}-g_k$

$$H_{k+1}^{DFP}=H_k+\frac{v_k v_k^T}{v_k^T y_k}-\frac{H_k y_k y_k^T H_k}{y_k^T H_k y_k}, \qquad \dots(31)$$

is the DFP formula

$$H_{k+1}^{BFGS}=H_k+\left[1+\frac{y_k^T H_k y_k}{v_k^T y_k}\right]\frac{v_k v_k^T}{v_k^T y_k}-\left[\frac{v_k y_k^T H_k+H_k y_k v_k^T}{v_k^T y_k}\right]; \quad ...(32)$$

all three forms satisfy the quasi-Newton condition $H_{k+1}y_k=v_k$ and maintain positive definite matrices if H_1 is positive definite. And we use the last one (BFGS) to update H_k in our new proposed algorithm and the scalar of HS to scale the direction d_k .

5.1 Preconditioned CG algorithm (PCG):

The preconditioned CG methods (PCG) first appeared in paper by Axelsson in 1972 , it was developed with object of accelerating the convergence of the CG-method by a transformation of variables while keeping the basic properties of the method. Such transformation was

introduced by Allwright in 1972, the symmetric positive definite matrix H can be factored in various ways for example as $H = LL^T$ where L is lower triangular and non-singular.

Buckly in 1978 [4] introduced an algorithm in which conjugate gradient and quasi-Newton search directions occur together and which can be extended this type of algorithms see for example [1].

The search direction to the preconditioned (PCG) method defined by:

$d_t = -H_k g_t$

$d_{k+1} = -H_k g_k + \beta_k d_k$ for $k \geq 1$

$$\beta_{HS} = \frac{g_{k+1}{}^T H\ y_k}{d_k{}^T H\ y_k} \qquad \ldots(33)$$

where H is the BFGS update defined as :

$$H_{i+1}^{BFGS} = H_i - \left[\frac{H_i y_t v_t^T}{v_t^T y_t}\right] + v_t \left[\frac{v_t^T y_t + y_t^T H_i y_t}{(v_t^T y_t)^2} - \frac{H_i y_t}{v_t^T y_t}\right]^T$$

For more detail we state the following algorithm (Buckely,1978):

Algorithm (Buckley, 1978)

Start with x_1, $H_1 = I$,let t = 1, i = 1, $\varepsilon > 0$.

Step 1:let $d_t = -H_i g_t$ if $\|g_t\| < \varepsilon$

Step2: $\forall$ k = t , t+1 , t+2,… repeat with $x_{k+1} = x_k + \lambda_k d_k$

$$d_{k+1} = -H_i g_{k+1} + \frac{y_k^T H_i y_k}{d_k{}^T y_k} d_k \text{ where } y_k = g_{k+1} - g_k$$

Step 3: we use the restarting when $\left|\frac{g_k^T H_i g_{k+1}}{g_{k+1}^T H_i g_{k+1}}\right| > 0.2$

when this condition is satisfied it restarts t to k and go to step 4

Step4: Update H_i by H_{i+1}(using BFGS formula) to get:

$$H_{i+1}^{BFGS} = H_i - \left[\frac{H_i y_t v_t^T}{v_t^T y_t}\right] + v_t \left[\frac{v_t^T y_t + y_t^T H_i y_t}{(v_t^T y_t)^2} - \frac{H_i y_t}{v_t^T y_t}\right]^T$$

Step 5: Replace i with i+1 and go to step1.

5.2 Self-Scaling PCG method:

Clearly self-scaling techniques are very effective in unconstrained optimization algorithms. In this section a new PCG method for solving unconstrained optimization problems is proposed.

This PCG algorithm considered here has an additional property of being invariant under scaling of the function is twice continuously differentiable and search direction is descent i.e $g_k^T d_k < 0$ also we assume that line search is exact i.e $g_k^T d_k = 0$

Let $d_{k+1} = -H_k g_{k+1} + \beta_k d_k$ ...(34)

where β_k as in (33) where the matrix H_k is an approximation of G^{-1} the inverse of Hessian of the objective function *f* (x). One important feature of PCG method is the choice of H_k the method requires H_k to be positive definite to deduce directions.

5.3 Theorem (New proof for the positive definite H_k in equation (34)):

Let *G* be a n×n symmetric matrix , let G v = y where $y_k = g_{k+1} - g_k$ and $v_k = x_{k+1} - x_k$ and let $m_0 = v_k^T v_k$, $m_1 = y_k^T v_k$, $m_2 = y_k^T y_k$ then the quantity

$$\alpha = \frac{m_1}{m_0}, \qquad \ldots.(35)$$

is an approximation for an eigen value λ of *G* and if we set $\alpha = \lambda + \varepsilon$,so that ε is the error of α ,then $|\varepsilon| \le \sqrt{\frac{m_2}{m_0} - \alpha^2}$, ...(36)

Proof:

Let δ^2 denote the radiant in (36) then since $m_1 = \alpha m_0$,we have

$$(y - \alpha v)^T (y - \alpha v) = m_2 - 2\alpha m_1 + \alpha^2 m_0 = m_2 - \alpha^2 m_0 = \delta^2 m_0$$

,.....(37)

Since *G* is symmetric, it has an orthogonal set of n real unit eigen vectors $z_1, z_2, \ldots\ldots\ldots\ldots z_n$ corresponding to the eigen values $\lambda_1, \ldots\ldots\ldots\ldots, \lambda_n$ respectively. (some of them may be equal).Then v has a representation of the form

$v = a_1 z_1 + \ldots + a_n z_n$ ...(38)

now $G z_1 = \lambda_1 z_1$,etc ,and we obtain y = G v $= a_1 \lambda_1 z_1 + \ldots + a_n \lambda_n z_n$ and , since the z_i are orthogonal unit vectors

, $m_0 = v^T v = a_1^2 + \ldots + a_n^2$ ...(39)

it follows that (37)

$y - \alpha v = a_1(\lambda_1 - \alpha) z_1 + \ldots + a_n(\lambda_n - \alpha) z_n$ since the z_i are orthogonal unit vectors we thus obtain from (37)

$\delta^2 m_0 = a_1^2(\lambda_1 - \alpha)^2 + ... + a_n^2(\lambda_n - \alpha)^2$

replacing each $(\lambda_i - \alpha)^2$ by the smallest of these terms, we have from (39)

$\delta^2 m_0 \geq (\lambda_c - \alpha)^2(a_1^2 + ... + a_n^2) = (\lambda_c - \alpha)^2 m_0$

Where λ_c is an eigen value to which α is closest .dividing this inequality by

m_0 and taking square root, we obtain (36). And the theorem is proved.

Because the matrix G is positive definite then all eigen values are positive. □

5.4 Theorem (New descent property):

The new search direction which is defined in

$$d_{k+1} = -g_{k+1} + \frac{v_k^T v_k}{v_k^T y_k}\frac{g_{k+1}y_k^T}{d_k^T y_k}.d_k, \qquad .....(40)$$

is always negative with exact line search.

Proof:

Multiplying (40) by g_{k+1} then we have

$$d_{k+1}^T.g_{k+1} = -g_{k+1}^T.g_{k+1} + \frac{v_k^T v_k}{v_k^T y_k}\frac{g_{k+1}y_k^T}{d_k^T y_k} d_k g_{k+1}^T$$

$$= -g_{k+1}^T.g_{k+1} + \frac{v_k^T v_k}{v_k^T y_k}\frac{g_{k+1}^T g_{k+1}}{d_k^T y_k} d_k y_k^T \quad = -g_{k+1}^T g_{k+1}(1 - \frac{v_k^T v_k}{v_k^T y_k})$$

Because $\frac{v_k^T v_k}{v_k^T y_k}$ is a $n \times n$ approximation for an eigen value of *H*, then

$\frac{v_k^T v_k}{v_k^T y_k}$>0 and always we have the direction d_{k+1} is negative.

5.5 New PCG- algorithm with dynamical retards:

Out lines of the new proposed PCG algorithm:

Let kret : number of retard .

knm : number of non-monotone .

Step1: Given x_1 , n is the dimension of the problem, k=1,Ac=1×10^{-4}

integer M > 0 , $0 < \delta < 1$, $H_0 = I$, $\gamma = 0.001$,knm = 0, kret = 0

Step 2: β=1, $d_k = -H_k g_k$, $z = f(x_k)$

Step 3: $Ar = 1$

Step 4: If (Ar >1) go to step 5(a) else $\lambda = \beta$ go to step 6

Step 5 (a): If (Ar > 2) go to step 5(b) else $\beta = 0.2$, $\lambda = \beta$ go to step 6

Step 5(b) : $\lambda = \beta^{Ar-1}$

Step 6: $x_{k+1} = x_k + \lambda_k d_k$, compute $f_{k+1} = f\left(x_{k+1}\right)$

Step 7: If $f_{k+1} - z \le 0.1.\lambda\,(g_k{}^T d_k - 0.5.\lambda.g_k{}^T d_k)$ go to step 8

else $Ar = Ar + 1$ go to step 4

Step 8: for j = 0, min (k ,M) find $\max f_{(k-j)} - \gamma.\lambda.g_k^T.g_k$

Step 9: if $f_{k+1} > \max f_{(k-j)} - \gamma.\lambda.g_k^T.g_k$ then $\lambda = \delta.\lambda$, knm = knm+1

$x_{k+1} = x_k + \lambda.d_k$ else kret = kret+1 $x_{k+1} = x_k + \lambda.d_k$

Step 10: $NOI = NOI + 1$

Step 11: If $\|g_{k+1}\| < AC$ then go to step 15 else continue.

Step 12: $\alpha = \frac{v_k^T y_k}{v_k^T v_k}$ Compute H_{k+1} by the BFGS update and compute the

scalar β_k^{HS} where $\beta_k^{HS} = \frac{g_{k+1}{}^T H\ y_k}{d_k{}^T H y_k}$

Step 13:Compute a new direction $d_{k+1} = -H_k g_{k+1} + 1/\alpha\beta_k^{HS} d_k$

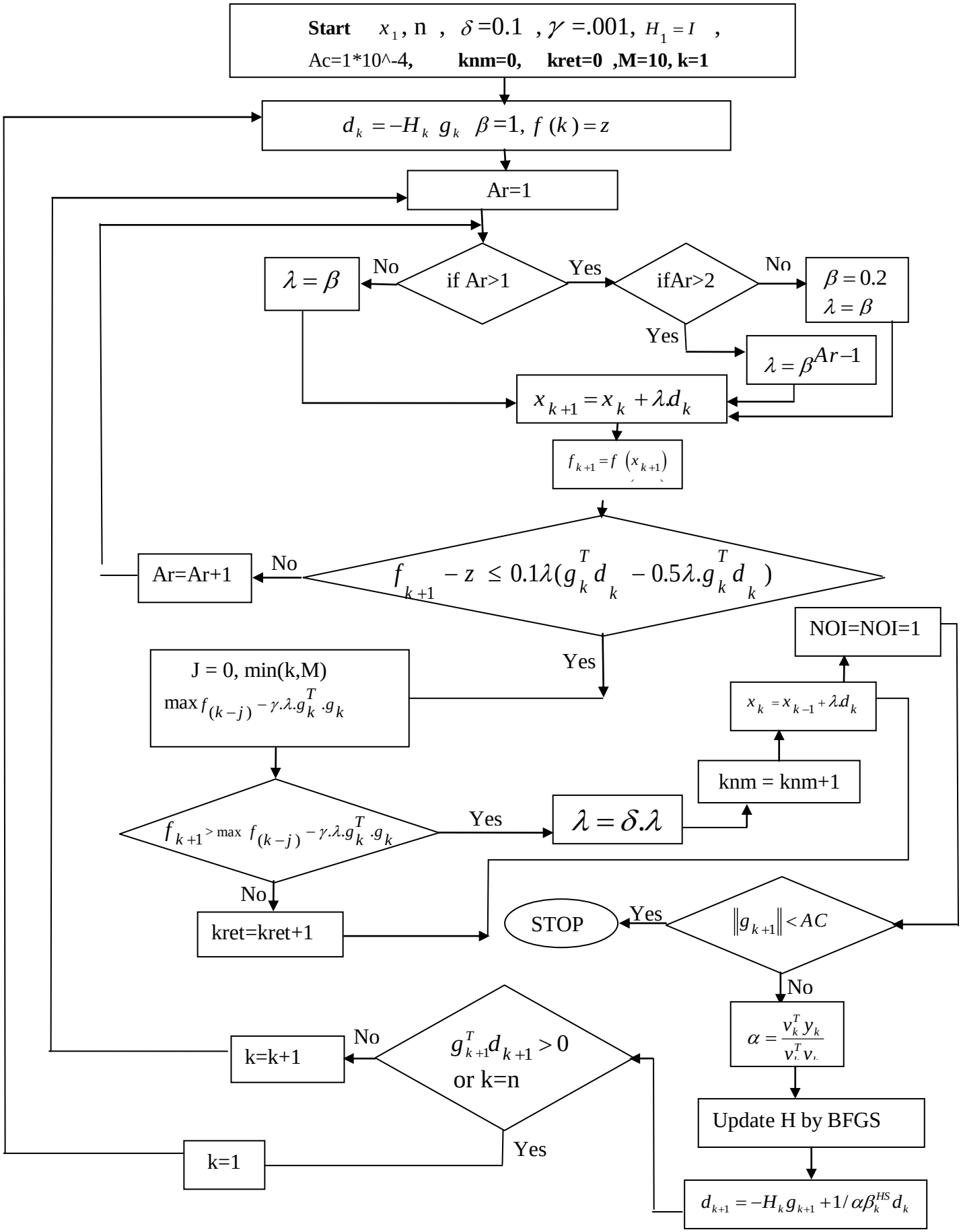
Step 14: If ($g_{k+1}{}^T d_{k+1} > 0$ or k=n) then k=1 and go to step 2 else k=k+1 go

to step 3

Step 15: End

And now we are going to represent the flowchart of the new proposed PCG- algorithm as follows:

5.6 Flow chart for the new proposed algorithm:



5.8 The Global Convergence Theorem (for the new proposed algorithm):

If $f(x_k) \le f_k^i \le \underset{0 \le j \le \min(k,M)}{\text{Max}} f(x_{k-j}) + \gamma g_k^T (x_{k+1} - x_k)$ .....(41)

and $f^i(x_k + \lambda_k d_k) = f(X_k)$ ....(42)

holds , then the algorithm is well defined in the sense that

$f(x_k + \lambda_k d_k) \le \underset{0 \le j \le \min(k,M)}{\text{Max}} f(x_{k-j}) + \gamma g_k^T (x_{k+1} - x_k)$ ....(43)

hold for finite (M).

Proof:

Assume by contradiction that, at the iteration k, the test (43) is never satisfied then there exist sequence $\{\lambda_M\}$,with $\lambda_M \longrightarrow 0$ as $M \longrightarrow \infty$ such that

$$f(x_k + \lambda_k d_k) \ge \max_{0 \le j \le \min(k,M)} f(x_{k-j}) + \gamma g_k^T (x_{k+1} - x_k), \quad \text{....(44)}$$

Remember now that, because of (41) we have that $f^i{}_k - f(x_k) \ge 0$ now if $f^i{}_k - f(x_k) > 0$ since $\lambda_M \longrightarrow 0$ for sufficiently large M ,(44) yields that $f^i{}_k - f(x_k) \le 0$,which is a contradiction, if instead $f^i{}_k - f(x_k) = 0$,then dividing both term of (44) by λ_M and taking the limit for $M \longrightarrow 0$ we obtain that $g_k^T d_k \ge \gamma g_k^T d_k$.

6. Numerical results and conclusions:

6.1 numerical results:

The two proposed algorithms describes in this paper namely:

1. The original non monotone algorithm (original)
2. The new proposed algorithm (new)

They are programmed in double precision (FORTRAN 95) .The complete set of results are given in Tables (6.1.1) - (6.1.4). The numerical comparison are the number of function evaluations NOF, number of iterations NOI; number of non monotones (NONM) and the number of dynamical retards (NODR) are considered. The actual convergence criterion employed was $\|g_{k+1}\| < 1*10\wedge -4$ for all the algorithms twenty well-known test functions with different dimensions are employed in this comparison.

The first table contains the results of ten test functions for n = 4 only, while the second table contains another ten test function of dimensionality one hundred and one thousand.

Table (6.1.1)
Comparison between the original & new algorithm
for n=4 only
(standard test functions)

	Origin				New			
	NOI	NOF	NONM	NODR	NOI	NOF	NONM	NODR
ROSEN	96	193	25	96	61	146	12	49
BEALE	40	81	8	40	13	34	0	11
DIXON	39	79	10	39	13	32	0	13
WOLFE	23	47	4	23	10	30	0	10
CANTRAL	59	119	12	59	37	82	0	37
SUM	14	29	0	14	16	36	0	16
WOOD	333	667	83	333	45	118	5	40
MILEL	181	363	49	181	28	62	0	28
CUBIC	159	319	42	159	76	184	19	57
POWELL	282	565	77	282	32	76	0	32
TOTAL	1226	2462	310	1226	331	800	36	293

From the above table it is clear that the new proposed algorithms has an improvements on the original non monotone algorithm (origin) by taking 100% [NOI ;NOF] for the origin algorithm we have in table (6.1.2)

Table (6.1.2)
Performance percentage of the new algorithm compared with the original algorithm

Tools	NOI	NOF
Origin	100%	100%
new	26%	32%

It is clear from the above table, for small dimensionality test function (n = 4)
and for ten selected test functions there are an important of 74% NOI ; 68% NOF .

Table (6.1.3)
Comparison between the (origin) and (new) algorithms
for dimensionality n=100 and n=1000

	N	Origin				New			
		NOI	NOF	NONM	NODR	NOI	NOF	NONM	NODR
MILEL	**100**	105	211	0	105	28	67	0	28
BEALE	**100**	49	99	0	99	19	53	0	16
DIXON	**100**	1492	2985	322	1492	160	495	114	46
WOLFE	**100**	44	89	2	44	110	422	66	44
CANTRAL	**100**	94	189	0	94	40	87	0	40
MILEL	**1000**	130	261	0	130	40	100	0	40
BEALE	**1000**	49	99	0	49	28	73	0	19
Shallo	**1000**	23	47	0	23	19	48	0	19
CANTRAL	**1000**	111	223	0	111	42	91	0	42
POWELL	**1000**	120	241	1	120	397	1426	91	306
TOTAL		2217	4444	325	2267	883	2862	271	600

From the above table it is clear that the new proposed algorithms has an improvements on the original Non-monotone algorithm (origin)by taking 100% [NOI ; NOF] for the origin algorithm we have in table (6.1.4)

Table (6.1.4)
Performance percentage of the new algorithm compared with the original algorithm

Tools	NOI	NOF
Origin	100%	100%
new	39%	64%

It is clear from the above table, for the dimensionality test function $100 \le n \le 1000$ there are an improvements of 61% NOI; 36% NOF.

6.2 Conclusions:

In this paper a new PCG-algorithm which employs a Non-monotone search direction with dynamical retards is investigated both theoretically and numerically.
The new proposed algorithm employs a modified line search sub program and it saves about 77% of NODR (Dynamical Retards) for small size of test functions while it saves a bout 74% of NODR for high-size test problems.

7. Appendix:

All the test function used in this paper is from general literature [2]:

1. Cube function,

$$F(x) = 100(x_2 - x_1^3)^2 + (1 - x_1)^2, \qquad x_0 = (-1.2, 1.0)^T$$

2. Beale function,

$$F(x) = (1.5 - x_1(1 - x_2))^2 + (2.25 - x_1(1 - x_2^2))^2 + (2.625 - x_1(1 - x_2^3))^2,$$
$$x_0 = (0,0)^T$$

3. Miele and Cornwell function,

$$F(x) = (e^{x1} - 1)^2 + \tan^4(x_3 - x_4) + 100(x_2 - x_3)^6 + x_1^8 + (x_4 - 1)^2,$$
$$x_0 = (1,2,2,2)^T$$

4. Dixon function,

$$F(x)=(1-x_1)^2+(1-x_{10})^2+\sum_{i=2}^{n}(x_i^2-x_{i+1})^2, \qquad x_0=(-1;...)^T$$

5. Shallow function,

$$F(x)=\sum_{i=1}^{n/2}[(x^{\,2}{}_{2i-1}-x_{2i})^2+(1-x_{2i-1})^2] \qquad x_0=(-2;...)^T$$

6. Powell function (generalized form),

$$F(x)=\sum_{i=1}^{n/4}[(x_{4i-3})+10x_{4i-2})^2+5(x_{4i-1}-x_{4i})^2+(x_{4i-2}-2x_{4i-1})^4+10(x_{4i-3}-x_{4i})^4]$$

, $x_0=(3,-1,0,1;...)^T$

7. Wood function (generalized form),

$$F(x)=\sum_{i=1}^{n/4}[100(x_{4i-2}-x_{4i-3}^2)^2+(1-x_{4i-3})^2+90(x_{4i}-x_{4i-1}^2)^4+(1-x_{4i-1})^2+$$

$$10.1(x_{4i-2}-1)^2+(x_{4i}-1)^2+19.8(x_{4i-2}-1)(x_{4i}-1)],x_0=(-3,-1,-3,-1;...)^T$$

8. Rosenbrock Banana function,

F(x)=100 $(x_2-x_1^2)^2+(1-x_1)^2, \qquad x_0=(-1.2,1.0)^T$

9. Wolfe function,

$$F(x)=(-x_1(3-x_1/2)+2x_2-1)^2+\sum_{i=1}^{n-1}(x_{i-1}-x_i(3-x_i/2)+2x_{i+1}-1)^2$$

$$+(x_{n-1}-x_n(3-x_n/2)-1)^2,x_0=(-1;...)^T$$

10. Sum function,

F(x)= $(x_1-1)^4, \qquad x_0=(2;...)^T$

11.Cantral function

F(x)=

$$\sum_{i=1}^{n/4}[(\exp(x_{4i-3})-x_{4i-2})^4+100(x_{4i-1}-x_{4i-1})^6+(\arctan(x_{4i-1}-x_{4i}))^4+x_{4i-3}]$$

$x_0=(1,2,2,2)^T$

8. References :

1.AL-Bayati A.Y ,(1996) "Anew PCG method for unconstrained Non-linear optimization", Iraq,vol(5),No.1,pp.71-92.

2.AL-Bayati A.Y , (2001),"New Generalized CG-methods for the Non-quadratic Model In unconstrained optimization" ,Journal of AL-yarmok , Jordan ,vol (10), pp 1-17.

3. Beal,E.M.L. (1972)."A Derivation of Conjugate Gradient". In: Numerical Methods For Nonlinear Optimization. Edited by Lootsma, F.A., Academic Press, pp.39-43.

4. Buckley, A. G. (1978). "A Combined Conjugate Gradient Quasi-Newton Minimization Algorithm" . Math.Prog.Vol.15, pp.200-210.

5. Dixon L.C.W ; spedicato E. and Szego G.P. (1980), "Non Linear Optimization Theory and Algorithms" , USA.

6.Fletcher, R. and reeves.C.M.(1964)."Function Minimization by Conjugate Gradients". Computer Journal, 7, pp.149-54.

7. Fletcher, R. (1987). "Practical Method of Optimization". John Wiley and Sons, Chichester, New York, ,Britain, Toronto ,and Singapore.

8.Grippo, F. Lampariello, and S.LUCID, (1986) , "Anonmonotone line search technique for Newton's method", SIAM J. Number. Anal., 23, pp.707-716.

9. Hestense ,M.R. and Stiefel, E. (1952). "Methods of Conjugate Gradient for solving Linear System". Journal Res .N.B.S., Vol.49, pp.449-464.

10.Martinez ,J.M. ,Pilotta,E.A., Raydan,M., (2003),"Spectral Gradient Methods for Linearly Constrained", special communication.

11. polak ,E. and Ribier G.,(1969), "Note Sur La Convergence Des Methods De Direction Conjugate " , Rev. Frinfr , Rech Oper.,16.R1.

12. Raydan ,F., (2003)"Gradient Method With Dynamical Retards For Large-Scale Optimization Problems", Kent state university,vol.16,pp.186-193.

13. Zhen-jun shi. and Jie shen,(2005) "step-size Estimation For Unconstrained Optimization Methods" university of Michigan ,Dearborn , MI 48128, USA, vol(24), N.3, pp.399-416.