A New Preconditioned Conjugate Gradient Method for Optimization

Issam A. R. Moghrabi, *Member, IAENG*

Abstract— **A new hybrid Conjugate Gradient (CG) method that generates weighted search directions is proposed in this paper. The weighting matrix is a two-step quasi-Newton update matrix that approximates the inverse Hessian matrix using data from the two most recent iterates. Multi-step methods have demonstrated substantial gains over the standard methods that utilize updates satisfying the classical Secant equation. The actual matrix for the method developed in this paper need not be explicitly retained. Instead, our implementation requires a few additional vectors that need to be updated at each iteration. The numerical performance of the new algorithm is then assessed by comparing it to the other methods developed in a similar context. A set of 1941 unconstrained optimization problems were used to test the algorithm. Our results reveal substantial improvements over the other two methods at a reasonable computational and storage costs. Those gains are particularly observed as the dimension of the problem increases.**

Index Terms— **Weighted Conjugate Gradient methods, quasi-Newton methods, multi-step methods, unconstrained optimization.**

I. Introduction

CONJUGATE gradient (CG) methods were originally devised for the solution of linear systems of equations. In addition to using the methods to to find the minimum point of a quadratic function, they can be used to minimize any unconstrained nonlinear continuous function of the form

$$minmize\ f(x), \qquad \text{where } f: R^n \to R,$$

for which the gradient can be computed. CG methods can be applied to a variety of optimization problems, such as nonlinear regression, neural net training and engineering design.

To minimize f, starting from an initial starting point x_0, the sequence of iterates $\{x_i\}$generated is given as

$$x_{i+1} = x_i + \alpha_i d_i, \qquad (1)$$

where α_i is a positive scalar that defines a step size on the CG search direction d_i. In nonlinear CG, the residual vector is taken as the negative of the gradient vector of the objective function. As with the linear CG, a value of α_i in (1) is computed such that the gradient is orthogonal to the search direction. Thus, the search direction d_i is computed using

$$d_i = \begin{cases} -g_i, & \text{for } i = 0, \\ -g_i + \beta_i d_{i-1}, & \text{for } i \geq 1, \end{cases} \qquad (2)$$

for some scalar β_i that is chosen to guarantee orthogonality and where g_i denotes the gradient of the function f evaluated at the point x_i. Normally, whenever β_i becomes negative, the method is restarted by setting $\beta_i = 0$ (see [29]), although other options have also been considered in the literature [2,515,17,21,29]. Some of those are adopted in our implementation and are mentioned later in this paper. The vector d_i is usually required to satisfy the condition

$$d_i^T g_i < 0,$$

to ensure it is downhill at x_i. It has been shown in several papers [5,21,23,28] that in order to guarantee global convergence, d_i may be required to satisfy the sufficient descent condition

$$d_i^T g_i \leq -\mathfrak{z}\|g_i\|^2,$$

for some constant $\mathfrak{z}$.

Different choices of β_i in (2) lead to different CG algorithms. Some well-known choices are

$$\beta_i^{FR} = \frac{\|g_i\|^2}{\|g_{i-1}\|^2}, \quad \beta_i^{PRP} = \frac{g_i^T(g_i - g_{i-1})}{\|g_{i-1}\|^2},$$
$$\beta_i^{HS} = \frac{g_i^T(g_i - g_{i-1})}{d_{i-1}^T(g_i - g_{i-1})}, \quad \beta_i^{LS} = \frac{g_i^T(g_i - g_{i-1})}{d_{i-1}^T g_{i-1}},$$
$$\beta_i^{DY} = \frac{g_i^T g_i}{d_{i-1}^T(g_i - g_{i-1})}.$$

The above methods are, respectively, due to Fletcher-Reeves [9], Polak–Ribiére–Polyak [22,23], Hestenes-Stiefel [16], Liu-Storey [17] and Dai–Yuan [5]. Many other choices have been considered in the literature (see, for example, [13,17,22,23,24]).

This paper derives a new CG algorithm that uses a weighted two-step update matrix in the computation of the search direction without having to explicitly retain the matrix in storage nor update it. The new method is inspired by the works of Anderi [2] and Ford et al. [13]. Anderi [2] applies updates to the identity matrix in order to build the conditioning matrix. Ford et al. [13] developed a multi-step CG method that does not directly involve any weighting matrix. Our derivation follows a hybrid approach that combines quasi-Newton and CG search directions. The conditioning matrix employed in our method exploits two-

Manuscript received January 17, 2018; revised July 25, 2018.

Issam A.R. Moghrabi is with the Gulf University for Science and Technology, Kuwait. He is currently a professor with the Department of Management Information Systems. phone: +965-94453637; e-mail: moughrabi.i@ gust.edu.kw).

step iteration information and the result is a descent CG method which utilizes multi-step quasi-Newton updates.

The next section summarizes the multi-step methods idea. Section 3 presents the derivation of the new method. Finally, the numerical results are summarized and conclusions presented.

II. Multi-step quasi-newton methods

Quasi-Newton methods retain an approximation to the Hessian matrix that is updated at each iteration to reflect the most recent changes in the data [3,25]. Given B_i, the current approximation to the Hessian $\nabla^2 f(x_{i+1})$, the new Hessian approximation, B_{i+1} is updated to satisfy the standard Secant equation:

$$B_{i+1} s_i = y_i, \tag{3}$$

where

$$s_i = x_{i+1} - x_i,$$

and

$$y_i = g_{i+1} - g_i.$$

A family of secant methods is the Broyden family [3,6] in which the updates are defined by

$$B_{i+1} = B_i - \frac{B_i s_i s_i^T B_i}{s_i^T B_i s_i} + \frac{y_i y_i^T}{s_i^T y_i} + \theta u_i u_i^T,$$

where θ is a scalar and

$$u_i = \sqrt{s_i^T B_i s_i}\left[\frac{y_i}{s_i^T y_i} - \frac{B_i s_i}{s_i^T B_i s_i}\right].$$

The BFGS, DFP and SR1 updates are obtained by setting $\theta = 0$, $\theta = 1$ and $= (1 - s_i^T B_i s_i / s_i^T y_i)^{-1}$, respectively.

The BFGS formula [3,10,11] is the most popular rank-two update formula that satisfies equation (3), especially that, as much of published research [3,13,15] have reported, it works well with inexact line search. The BFGS update to the Hessian matrix approximation is given by

$$B_{i+1} = B_i - \frac{B_i s_i s_i^T B_i}{s_i^T B_i s_i} + \frac{y_i y_i^T}{s_i^T y_i}.$$

In the standard Secant equation (3), a straight line L is used to find a new iterate x_{i+1}, given the previous iterate x_i, while in the multi-step methods higher order polynomials are employed, as will be shown next.

Let $\{x(\tau)\}$ or X denote a differentiable path in R^n, where $\tau \in R$. The vector polynomial $x(\tau)$ thus satisfies

$$x(\tau_j^{(i-1)}) = x_{i-m+j}, \text{ for } j = 0, 1, \ldots, m,$$

for some distinct values $\left\{\tau_j^{(i-1)}\right\}_{j=0}^{m}$. The corresponding gradient points are interpolated by a similar polynomial $z(\tau)$ satisfying

$$z(\tau_j^{(i-1)}) = g_{i-m+j}, \text{ for } j = 0, 1, \ldots, m.$$

Then upon applying the Chain rule to the gradient vector $g(x(\tau))$ in order to find the derivative of the gradient g with respect to τ, we get

$$\frac{dg}{d\tau} = G(x(\tau))\frac{dx}{d\tau} \quad . \tag{4}$$

Thus, at any point on the path X, the Hessian G must satisfy (4) for any value of τ. More specifically, for $\tau = \tau_c$, where $\tau_c \in \mathcal{R}$. This will result in the following relation

$$\left.\frac{dg}{d\tau}\right|_{\tau=\tau_c} = G(x(\tau))\left.\frac{dx}{d\tau}\right|_{\tau=\tau_c}$$

Thus, at any point on the path X, the Hessian G must satisfy (4) for any value of τ. More specifically, for some $\tau = \tau_c$, where $\tau_c \in R$ the resulting relation is given by

$$\left.\frac{dg}{d\tau}\right|_{\tau=\tau_c} = G(x(\tau))\left.\frac{dx}{d\tau}\right|_{\tau=\tau_c}.$$

By analogy with the Secant equation, the aim is to derive a relation satisfied by the Hessian at the new iterate $x_{i+1.}$ We choose a value for the parameter τ, namely $\tau_{m,}$, that corresponds to the most recent iterate as follows

$$g'(\tau_m) = B_{i+1} x'(\tau_m)$$

or

$$w_i = B_{i+1} r_i, \tag{5}$$

where the vectors r_i and w_i are defined in terms of the m most recent step vectors $\{s_k\}_{k=i-m+1}^{i}$ and the m most recent gradient difference vectors $\{y_k\}_{k=i-m+1}^{i}$ respectively, as follows

$$r_i = \sum_{j=0}^{m-1} s_{i-j}\left\{\sum_{k=m-j}^{m} L'_k(\tau_m)\right\}$$

and

$$w_i = \sum_{j=0}^{m-1} y_{i-j}\left\{\sum_{k=m-j}^{m} L'_k(\tau_m)\right\}$$

for

$$L'_k(\tau_m) = (\tau_k - \tau_m)^{-1}\left[\frac{\tau_m - \tau_j}{\tau_k - \tau_j}\right], k < m$$

and

$$L'_m(\tau_m) = \sum_{j=0}^{m-1} (\tau_m - \tau_j)^{-1}$$

are the standard Lagrange polynomials.

Ford and Moghrabi [10,11,12,18,19] examined several choices for the parameters $\{\tau_k\}_{k=0}^{m}$. It was found that different choices influence the structure of the interpolating curve and consequently the numerical performance of the multi-step methods.

The choices made for the parameters $\{\tau_k\}_{k=0}^{m}$ rely on some metric of the following form

$$\emptyset_M(z_1, z_2) = [(z_1 - z_2)^T M (z_1 - z_2)]^{1/2},$$

where M is a symmetric positive-definite matrix. The quantity $\emptyset_M(z_1, z_2)$ defines the distance between any two vectors z_1 and z_2, using some metric. For example, if $M = I$, then $\emptyset_M(z_1, z_2)$ denotes the 2-norm.

Of the approaches considered in [12], the most numerically successful choice is selected in this work to derive our new algorithm. The choice is based on the socalled Accumulative Approach. The Accumulative approach chooses one of the iterates, say x_j, as a base-point and sets the parameter τ_j corresponding to it to 0. Then, any value τ_k, corresponding to the point $x_{i-m+k+1}$ for any k except for $k=j$, is computed by distance accumulation (measured by the chosen metric Φ_M) between each two consecutive pair of points in the sequence from $x_{i-m+j+1}$ to $x_{i-m+k+1}$. Therefore, any value τ_k, for $k = 0,1,\ldots,m$, is obtainable using

$$\begin{aligned}\tau_k &= -\textstyle\sum_{p=k+1}^{j} \phi_M\left(x_{i-m+p+1}, x_{i-m+p}\right), k < j,\\ &= 0,\ k = j,\\ &= -\textstyle\sum_{p=j+1}^{k} \phi_M\left(x_{i-m+p+1}, x_{i-m+p}\right), k > j. \qquad (6)\end{aligned}$$

This approach will yield values of τ that satisfy

$$\tau_k < \tau_{k+1}, for\ k = 0, 1, \ldots, m-1,$$

under the assumption that no two consecutive points overlap.

Those values of the parameters $\{\tau_k\}$ are the ones used in computing the vectors $x'(\tau_m)$ and $g'(\tau_m)$ in (5) (or, equivalently, vectors r_i and w_i, respectively) needed to compute the new Hessian approximation B_{i+1}. It should be noted that different choices of the metric matrix M in ϕ_M will result in different methods. Choices investigated for M (see [11,12,18,19]), include $M = I$, $M = B_i$, and $M = B_{i+1}$. The method derived in this paper uses the choice $M = I$.

Ford and Moghrabi [11,12] indicate that values of $m > 2$ do not yield further substantial numerical gains in performance due to the non-smoothness of the interpolant. Thus, $m = 2$ is chosen here and such methods are termed two-step methods as they utilize data from the two most recent iterations to update the Hessian approximation.

The inverse Hessian approximation update generally satisfies:

$$H_{i+1}(y_i - \mu_{i-1} y_{i-1}) = s_i - \mu_{i-1} s_{i-1} \qquad (7)$$

Or, equivalently

$$w_i = B_{i+1} r_i$$

where

$$\mu_{i-1} = \frac{\delta_{i-1}^2}{2\delta_{i-1} + 1}.$$

and

$$\delta_{i-1} = \frac{\tau_2^{(i-1)} - \tau_1^{(i-1)}}{\tau_1^{(i-1)} - \tau_0^{(i-1)}}.$$

For $M = I$ in (6), the corresponding τ-values are given by

$$\tau_0 = -(\|s_i\|_2 + \|s_{i-1}\|_2), \tau_2 = 0, \text{and } \tau_1 = -\|s_i\|_2 .$$

This, hence, gives a value for δ in (7) as

$$\delta = \frac{\|s_i\|}{\|s_{i-1}\|}. \qquad (8)$$

Equation (8) may be generalized by introducing a scaling factor, $\gamma \geq 0$ (see [13]) that provides more control and is exploited for convenient switching to the standard one-step Secant equation update method, simply by setting the scalar to zero. Therefore,

$$\delta = \gamma \frac{\|s_i\|}{\|s_{i-1}\|}. \qquad (9)$$

The multi-step B-version BFGS formula is given by

$$B_{i+1}^{MS} = B_i + \frac{w_i w_i^T}{w_i^T r_i} - \frac{B_i r_i r_i^T B_i}{r_i^T B_i r_i}. \qquad (10)$$

III. A NEW MULTI-STEP PRECONDITIONED CG METHOD (MSPCG)

The approach considered here is based on the preconditioning of the gradient in (2) in order to improve the convergence of the CG method. If the initial direction d_0 is selected as $d_0 = -g_0$ and the objective function to be minimized is a convex quadratic function in the form of

$$f(x) = \frac{1}{2} x^T A x + b^T x + c,$$

and exact line searches are used, that is

$$\alpha_i = arg\ min_{\alpha>0} f(x_i + \alpha d_i),$$

then the conjugacy condition is satisfied

$$d_i^T A d_j = 0, \forall i \neq j, \qquad (11)$$

where A is the positive definite Hessian matrix of f. Then, for general nonlinear twice differentiable function f, and using the mean value theorem, there exists some $\omega \in (0,1]$ such that

$$d_i^T y_{i-1} = \alpha_{i-1} d_{i-1} \nabla^2 f(x_{i-1} + \omega \alpha_{i-1} d_{i-1}) d_{i-1}.$$

Thus, from (11), it makes sense to replace the quadratic conjugacy condition (11) with the following alternative

$$d_i^T y_{i-1} = 0. \qquad (12)$$

Perry [21] extended the conjugacy condition (11) by utilizing second-order information. For the Hestenes-Stiefel conjugate gradient algorithm with $\beta_i^{HS} = \frac{g_i^T(g_i - g_{i-1})}{d_{i-1}^T(g_i - g_{i-1})}$ (see [25]), Perry notes that the search direction (3) can be expressed as

$$d_{i+1} = -\left[I - \frac{s_i y_i^T}{y_i^T s_i}\right] g_{i+1} = -Q_{i+1}^{HS} g_{i+1}.$$

The matrix Q_{k+1}^{HS} resembles an approximation to the inverse Hessian though it is neither positive definite nor symmetric although the search direction obtained, namely, d_{i+1}, satisfies the conjugacy condition (11). Perry [21] notes that, when the line searches are not exact, it is more appropriate to choose the approximation to the inverse Hessian to satisfy the Secant condition (3) than the conjugacy condition (11). Therefore, the following equality is used instead in [21]

$$-g_{i+1} + \beta_i s_i = -B_{i+1}^{-1} g_{i+1}$$

where B_{i+1} is as in (3). Consequently, Perry's choice for β_i is given by

$$\beta_i = \frac{y_i^T g_{i+1} - s_i^T g_{i-1})}{s_i^T y_i}.$$

The corresponding search direction is computed using

$$d_{i+1}=-\left[I-\frac{s_i y_i^T}{y_i^T s_i}+\frac{s_i s_i^T}{y_i^T s_i}\right] g_{i+1}=-Q_{i+1}^P g_{i+1}.$$

If the line search direction is exact, than Perry's algorithm is equivalent to the HS conjugate gradient method and satisfies the Dai and Yuan [5] conjugacy condition

$$d_{i+1}^T y_i=-u(s_i^T g_{i+1}),$$

with $u=1$.

The above argument motivates our choice of the search direction in this paper which takes the general form

$$d_i = -\sigma_i g_i + \beta_i s_{i-1}, \tag{13}$$

where σ_i can be chosen to be some scalar or positive definite matrix. For example, if $\sigma_i = 1$, then (13) is equivalent to (2). If, however, σ_i is chosen to be some approximation to the inverse of the Hessian matrix, then d_i becomes a combination of the quasi-Newton and the conjugate gradient directions, as will be the case in the derivation to follow. The main feature of our method is that σ_i is chosen to be a conditioning matrix that exploits second order and two-step iteration information that is expected to improve the quality of the generated search direction. The result is a descent Conjugate Gradient method with multi-step quasi-Newton updates.

Following Perry's argument [21] and given that the quasi-Newton search direction is given by $d_i = -H_i g_i$, the equality in (12) can be expressed by

$$d_i^T y_{i-1} = -g_i^T s_{i-1}. \tag{14}$$

Now, using (3) and (14), we obtain

$$d_i^T y_{i-1} = -g_i^T (H_i y_{i-1}).$$

From (7), we have

$$d_i^T y_{i-1} = -g_i^T r_{i-1} - \mu_{i-1} g_i^T H_i y_{i-2}.$$

This yields

$$d_i^T w_{i-1} = -\epsilon g_i^T r_{i-1}, \tag{15}$$

for some $\varepsilon \geq 0$ that serves as a scaling factor to impose conjugacy.

From (13) and (15), we have

$$-\sigma_i w_{i-1}^T g_i + \beta_i w_{i-1}^T s_{i-1} = -\epsilon g_i^T r_{i-1}, \tag{16}$$

which yields an expression for β_i as follows

$$\beta_i = \frac{g_i^T[\sigma_i w_{i-1} - \epsilon r_{i-1}]}{s_{i-1}^T w_{i-1}}. \tag{17}$$

If $\sigma_i = H_i$ and $\varepsilon = 1$, β_i vanishes and hence (17) reduces to the plain multi-step method search direction that satisfies the relation in (7). If $\varepsilon = 0$, then (17) reduces to the choice of β_i obtained in [13].

We proceed with our derivation with the choice $\sigma_i = H_i$. To complete the implementation details of the algorithm, the quantity (see (13))

$$d_{i+1} = -H_{i+1} g_{i+1} + \beta_{i+1} s_i,$$

needs to be computed efficiently without requiring that the update matrix is stored in explicit format. In specific,

$$z_{i+1} = H_{i+1} g_{i+1} \tag{18}$$

needs to be computed without having to store the matrix H_{i+1} or having to carry out any matrix-vector multiplication, thus adhering to the spirit of the CG methods.

The H-version of the multi-step BFGS formula, is given by

$$H_{i+1} = H_i - \frac{r_i w_i^T H_i + H_i w_i r_i^T}{w_i^T r_i} + \left(1 + \frac{w_i^T H_i w_i}{w_i^T r_i}\right)\frac{r_i r_i^T}{w_i^T r_i}. \tag{19}$$

It follows that

$$H_{i+1} g_{i+1} = H_i g_{i+1} - \frac{r_i^T g_{i+1}}{w_i^T r_i} v_i + \left[\left(1 + \frac{w_i^T v_i}{w_i^T r_i}\right)\frac{(r_i^T g_{i+1})}{w_i^T r_i} - \frac{(v_i^T g_{i+1})}{w_i^T r_i}\right] r_i, \tag{20}$$

where

$$v_i \cong H_i w_i.$$

Using (19), we obtain an expression for v_i in (20) as follows

$$v_i = w_i - \frac{r_{i-1}(w_{i-1}^T w_i) + w_{i-1}(r_{i-1}^T w_i)}{w_{i-1}^T r_{i-1}} + \left(1 + \frac{w_{i-1}^T w_{i-1}}{w_{i-1}^T r_{i-1}}\right)\frac{r_{i-1}(r_{i-1}^T w_i)}{w_{i-1}^T r_{i-1}}, \tag{21}$$

and

$$H_i g_{i+1} = v_i + \beta_i s_{i-1} - d_i + \mu_{i-1} H_i y_{i-1},$$

where μ_{i-1} is as in (7). To complete the derivation, we still need to be able to compute the quantity $H_i y_{i-1}$ efficiently. Using (3), we have

$$H_i y_{i-1} = H_i g_i - H_i g_{i-1} = z_i - H_i g_{i-1},$$

where z_i is as in (18). From (19), we get

$$H_i g_{i-1} = z_{i-1} - \frac{r_{i-1}(w_{i-1}^T z_{i-1}) + v_{i-1}(r_{i-1}^T g_{i-1})}{w_{i-1}^T r_{i-1}} + \left(1 + \frac{w_{i-1}^T v_{i-1}}{w_{i-1}^T r_{i-1}}\right)\frac{r_{i-1}(r_{i-1}^T g_{i-1})}{w_{i-1}^T r_{i-1}}. \tag{22}$$

Hence, the derivation is complete.

Al-Baali [1] proved global convergence of the Fletcher-Reeves method on general functions with inexact line searches. Similarly, Dai and Yuan [5] developed a CG method that is based on the Secant condition and also proved global convergence of their method. In order to guarantee the convergence of both the algorithms in [1] and [5], the step size α_i in (1) should satisfy the Wolfe conditions [31] (see [1,2,5,15]):

$$f(x_i + \alpha_i d_i) - f(x_i) \leq \rho_1 \alpha_i d_i^T g_i, \tag{23}$$

$$g(x_i + \alpha_i d_i)^T d_i \geq \rho_2 d_i^T g_i, \tag{24}$$

where $0 < \rho_1 \leq \rho_2 < 1$.

We now present the following theorem that states the conditions that ensure the search direction is downhill and hence guarantee convergence by the argument presented in [1,2,5,29].

Theorem 1. Suppose that α_i in (1) satisfies the Wolfe conditions (23) and (24); If $w_i^T r_i > 0$, then $-H_{i+1} g_{i+1}$ in (20) is a descent direction.

Proof. Given that $d_0 = -g_0$, it follows that $g_0^T d_0 = -\|g_0\|^2 \leq 0$. For subsequent iterations,

$$H_{i+1} g_{i+1} = H_i g_{i+1} - \frac{r_i^T g_{i+1}}{w_i^T r_i} v_i + \left[\left(1 + \frac{w_i^T v_i}{w_i^T r_i}\right) \frac{(r_i^T g_{i+1})}{w_i^T r_i} - \frac{(v_i^T g_{i+1})}{w_i^T r_i}\right] r_i.$$

Pre-multiplying (20) by $-g_{i+1}^T$ gives

$$-g_{i+1}^T H_{i+1} g_{i+1} = \frac{1}{(w_i^T r_i)^2} \left[\begin{matrix} -g_{i+1}^T H_i g_{i+1} (w_i^T r_i)^2 + 2(r_i^T g_{i+1})(w_i^T r_i)(g_{i+1}^T v_i) \\ -(g_{i+1}^T r_i)^2 (w_i^T r_i) - (r_i^T g_{i+1})^2 (w_i^T v_i) \end{matrix}\right].$$

If the inequality $u^T q \leq \frac{1}{2}(\|u\|^2 + \|q\|^2)$ is applied to the second term above with $u = (w_i^T r_i) g_{i+1}$ and $q = (r_i^T g_{i+1})\, v_i$, we obtain

$$-g_{i+1}^T H_{i+1} g_{i+1} \geq -\frac{(r_i^T g_{i+1})^2}{w_i^T r_i}.$$

Since $(r_i^T g_{i+1})^2 > 0$, then if $w_i^T r_i > 0$, it follows that $g_{i+1}^T H_{i+1} g_{i+1}$ is positive and the direction $-H_{i+1} g_{i+1}$ is a descent one for all i. Hence, the proof is complete.

Numerically speaking, the condition $w_i^T r_i > 0$ is checked every iteration to ensure it holds. Due to the approximation $v_i \cong H_i w_i$ used in (21), the condition might fail to hold. In that case, we resort to using $v_i = w_i$. By the above theorem and for a line search parameter α_i that satisfies conditions (23) and (24), convergence is guaranteed. If the function f is strongly convex and Lipschitz continuous on the level set $L_0 = \{x \in R^n : f(x) < f(x_0)\}$, then it becomes easy to prove that if the search direction given by (13) with α_i in (1) is computed to satisfy the Wolfe conditions (23) and (24), then it is the case that $g_i = 0$ for some i, or $lim_{i \to \infty}\{g_i = 0\}$. Theorem 1 and the restart procedure implemented both guarantee the convergence of the algorithm on general functions. So, under reasonable assumptions, the Wolfe relations and the Powell restart criterion (25) provide sufficient conditions to prove the global convergence of the algorithm.

We now present the algorithm steps as follows:

Algorithm MSPCG

```
Step 1. Choose the initial starting point $x_0 \in R^n$ and compute the gradient $g(x_0)$ and $d_0 = -g(x_0)$; set the iteration counter $i = 0$; select a value for $\varepsilon$ in (17). Set the stopping criteria parameter $\epsilon$.
Step 2. If $\|g_i\| < \epsilon$, then stop; otherwise go to step 3.
Step 3. Using the Wolfe line search conditions (23) and (24), determine the step length $\alpha_i$.
Step 4. $x_{i+1} = x_i + \alpha_i d_i$
Step 5. Compute: $g(x_{i+1})$, $s_i = x_{i+1} - x_i$, $y_i = g_{i+1} - g_i$, $w_i = y_i - \mu_{i-1} y_{i-1}$, and $r_i = s_i - \mu_{i-1} s_{i-1}$ (for $\mu_{i-1} = \frac{\delta_{i-1}^2}{2\delta_{i-1}+1}$) [see (8) and (9)].
Step 6. If $w_i^T r_i > 0$, then compute $v_i$ in (20) using (21) else set $v_i = w_i$.
Step 7. If the restart criterion of Powell (17) is satisfied, then set $d_{i+1} = -g_{i+1}$ else compute $d_{i+1} = -z_{i+1} + \beta_{i+1} s_i$, using the recurrence in (18)-(22).
Step 8. i++.
Step 9. Go to step 2. □
```

IV. Numerical Computations

The numerical experiments were conducted on 32 different test problems with dimensions varying from 2 to 100000. The unconstrained optimization test problems used in our experiments are listed in Table I and are primarily found in [14] and [20]. A good portion of the test problems have been selected from the updated CUTEr unconstrained problem collection [14]. The collection has been updated with almost 200 new examples since its release. These include large collections of problems arising from real-life quadratic programming problems and linear complementarity. Each problem was tested on six different starting points. The total number of test problems obtained is 1941. The functions tested are grouped into four categories, low (2 ≤ n ≤20), medium (21 ≤ n ≤ 40), moderately high (41 ≤ n ≤1000) and high (n >10000). Our computational results are benchmarked against Anderi's [2] SCALCG and the algorithm developed in [13]. Unlike the method in [13], Anderi's [2] and our algorithm both utilize a weighting matrix in the computation of the CG direction vector. The results reported in Table II correspond to the collective totals obtained on all the test problems. The scores reported indicate iteration, function/gradient evaluations as well as the total execution times. In addition, the right most column (Scores) indicate the number of times a method outperformed the other two methods on a specific problem. In case of a tie, the methods involved in that tie all get a score of one. Tables III to VI report the results for each of the four dimension categories mentioned earlier (high, moderately high, medium and low). The parameter γ in (9) was chosen to be one in the results reported here as values other than zero, such as 0.5 and 1.2, did not introduce any significant changes in the outcomes. The coding was done using C++ on a 64-bit machine with i7-3770, 3.4 GHZ CPU.

TABLE I
TEST PROBLEMS

Function	Dimension
Watson function [20]	3 ≤ n ≤ 14
Extended Rosenbrock [20]	2 ≤ n ≤ 10000, n even
Extended Powell [20]	2 ≤ n ≤ 10000, n div.by 4
Penalty function I [20]	2 ≤ n ≤ 1000
Variably dimensioned function [20]	2 ≤ n ≤ 10000
Trigonometric function [20]	2 ≤ n ≤ 10000
Modified Trigonometric function [14]	2 ≤ n ≤ 100000
Broyden Tridiagonal function [20]	2 ≤ n ≤ 1000
Discrete Boundary value function [20]	2 ≤ n ≤ 1000
Oren & Spedicato Power function [20]	2 ≤ n ≤ 10000
Distinct EigenValues Problem [20]	2 ≤ n ≤ 10000
Tridiagonal function [20]	2 ≤ n ≤ 10000
Wolfe function [20]	2 ≤ n ≤ 1000
Diagonal Rosenbrock's function [14]	2 ≤ n ≤ 1000, n even
Generalized Shallow function [14]	2 ≤ n ≤ 1000, n even
Powell Singular [20]	n = 10000
Sphere function [14]	n = 100
Trid function [14]	n = 100
Power Sum [20]	2 ≤ n ≤ 10000, n even
Dixon-Price [14]	2 ≤ n ≤ 10000, n even
Sum Squares [14]	2 ≤ n ≤ 10000
Broyden Banded [14]	2 ≤ n ≤ 10000
Penalty function II [20]	2 ≤ n ≤ 10000
Brown almost linear [14]	2 ≤ n ≤ 10000
Discrete Integral Equation [14]	2 ≤ n ≤ 10000
Kowalik and Osbome [14]	2 ≤ n ≤ 10000
Meyer [14]	2 ≤ n ≤ 10000
Bard [14]	2 ≤ n ≤ 10000
Linear full rank [14]	2 ≤ n ≤ 100000
Brown badly scaled [14]	2 ≤ n ≤ 10000
Linear rank 1 [14]	2 ≤ n ≤ 10000
Gulf Research & develop. [14]	2 ≤ n ≤ 10000

TABLE II
OVERALL ITERATION, FUNCTION EVALUATIONS COUNT AND TIMING

Method	Evaluations	Iterations	Time (sec.)	Scores
Ford et al.	355889 (88%)	221256 (89.2%)	44573.2 (89.1%)	554
Anderi's	407688 (100%)	249127 (100%)	49988.3 (100%)	365
MSPCG	**289532 (71.0%)**	**175898 (70.6%)**	**37286.1 (74.4%)**	**1022**

TABLE III
ITERATION, FUNCTION EVALUATIONS COUNT AND TIMING-LARGE PROBLEMS

Method	Evaluations	Iterations	Time (sec.)	Scores
Ford et al.	171118 (87.4%)	109311 (89.3%)	3678.8 (89.8%)	188
Anderi's	197955 (100.0%)	123011 (100.0%)	4112.7 (100.0%)	102
MSPCG	**140118 (70.3%)**	**86039 (69.5%)**	**3041.7 (73.1%)**	**331**

TABLE IV
ITERATION, FUNCTION EVALUATIONS COUNT AND TIMING- MODERATELY HIGH PROBLEMS

Method	Evaluations	Iterations	Time (sec.)	Scores
Ford et al.	118412 (87.5%)	70019 (89.4%)	8511.78 (86.8%)	171
Anderi's	135411 (100.0%)	78388 (100.0%)	9855.6 (100.0%)	131
MSPCG	**94612 (69.7%)**	**55617 (70.1%)**	**6911.4 (70.3%)**	**330**

TABLE V
ITERATION, FUNCTION EVALUATIONS COUNT AND TIMING- MEDIUM PROBLEMS

Method	Evaluations	Iterations	Time (sec.)	Scores
Ford et al.	54518 (88.7%)	33311 (86.0%)	20891.1 (89.0%)	128
Anderi's	61811 (100.0%)	38717 (100.0%)	23611.7 (100.0%)	91
MSPCG	**44211 (70.6%)**	**26977 (69.6%)**	**17141.3 (72.5%)**	**268**

TABLE VI
ITERATION, FUNCTION EVALUATIONS COUNT AND TIMING- SMALL PROBLEMS

Method	Evaluations	Iterations	Time (sec.)	Scores
Ford et al.	11841 (95.3%)	8611 (96.0%)	11490.7 (93.3%)	67
Anderi's	12511 (100.0%)	9011 (100.0%)	12408.3 (100.0%)	41
MSPCG	**10691 (85.0%)**	**7211 (80.2%)**	**10191.1 (82.1%)**	93

The numerical evidence, reported in Tables II to VI, reveals that the new method MSPCG shows substantial improvements over Anderi's [2] and over the method of Ford et al. [13] on the majority of the test problems. Compared to the algorithm in [2], the new method improves by an average of 29% overall and around 17% over the method in [13]. It has been observed that the merits of the new algorithm become more significant as the size of the problem increases. There are a few test problems on which either of the two other methods failed to converge (or took too long to do so) and on which our algorithm was able to reach a near optimal solution. Such failures were not included in the results reported in the above tables to maintain consistency of the benchmarking.

All methods use exactly the same line search implementation with choices $\rho_1 = 0.0001$ and $\rho_1 = 0.88$ in (23) and (24). The termination condition used is

$$\|g(x_i)\| \le 10^{-5}.$$

The methods were restarted periodically using Powell's restart test to measure the degree of orthogonality (see [22,26,27,30])

$$|g_{i+1}^T g_k| \ge 0.2\|g_{i+1}\|^2. \qquad (25)$$

We used Anderi's [2] restart search direction for SCALCG. As for MSPCG, the restart was done using $\beta_i = 0$ and $H_i = \sigma_i I$ in (16), for $\sigma_i = \frac{s_i^T s_i}{s_i^T y_i}$ (see [2,26,29]). This situation was not been encountered very frequently in our tests. Wolfe's conditions (23) and (24) ensure that $s_i^T y_i > 0$ and hence guarantee that Anderi's SCALCG [2] search direction is downhill. In our case, by analogy, $r_i^T w_i > 0$ ensures that the search direction is descent. Nevertheless, due to the approximation used in (20), numerical safeguarding remains a must [24].

As a further means of assessment, We have introduced the CPU performance profile as was done in Anderi [2] as shown in the figure below.

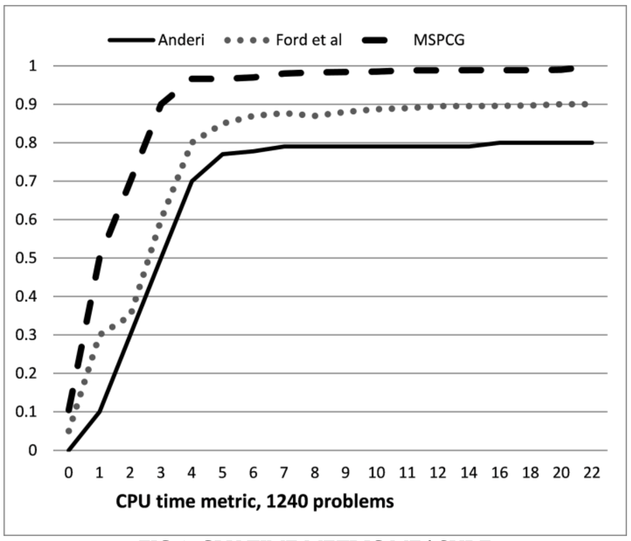

FIG 1. CPU TIME METRIC MEASURE

V. Conclusion

In this paper, a new weighted descent Conjugate Gradient method with multi-step quasi-Newton updates is developed. The method generates search directions that are a combination of the two-step quasi-Newton and CG vectors in an attempt to utilize the advantages of both methods. The derived method requires a few additional vectors to be retained and updated, compared to SCALCG and the method in [3]. The reasonable savings in computational costs, especially on large problems, seem to present a good justification for that extra storage requirement. The conditioning matrix utilized in our method exploits iteration information accumulated from the two latest steps.

We are currently investigating other choices for the weighting matrix to determine whether the numerical performance of similar methods can be improved further. There also remains the issue of developing automatic restart criteria that provides appropriate switching among several options similar to what was done in [1]. The global convergence properties of such methods are also under consideration.

References

[1] Al-Baali, M. "New property and global convergence of the Fletcher–Reeves method with inexact line searches", *IMAJ. Numer. Anal.,* 5, Feb. 1985, pp. 122–124.

[2] Anderi, N. "A scaled BFGS preconditioned conjugate gradient algorithm for unconstrained optimization", *Applied Mathematics Letters*, 20, April 2007, pp. 645-650.

[3] Broyden, C.G. "The convergence of a class of double-rank minimization algorithms - Part 2: The new algorithm", *J. Inst. Math. Applic.*, 6, June 1970, pp. 222-214.

[4] Byrd, R.H., Schnabel, R.B., and Shultz, G.A., "Parallel quasi-Newton methods for unconstrained optimization", *Math. Programing*, 42, April 1988, pp. 273-306.

[5] Dai, Y.H. and Yuan Y., "A nonlinear conjugate gradient method with a strong global convergence property", *SIAM J. Optim.*, 10, Dec. 1999, pp. 177–182.

[6] Dennis, J.E. and Schnabel, R.B., "Least change Secant updates for quasi-Newton methods", *SIAM Review*, 21, Feb. 1979, pp. 443-459.

[7] Fletcher, R., *Practical Methods of Optimization* (second edition), Wiley, New York, 1987.

[8] Fletcher, R., "A new approach to variable metric algorithms", *Comput. J.* , 13, April 1970, pp. 147-322.

[9] Fletcher, R. and Reeves, C., "Function minimization by conjugate gradients", *Computer J.*, 7, Jan. 1964, pp. 149–154.

[10] Ford, J.A. and Moghrabi I.A.R., "Using function-values in multi-step quasi-Newton methods", *J. Comput. Appl. Math.*, 66, March 1996, pp. 201-211.

[11] Ford, J.A. and Moghrabi I.A.R., "Multi-step quasi-Newton methods for optimization", J. *Comput. Appl. Math*., 50, Jan. 1994, pp. 305-323.

[12] Ford, J.A. and Moghrabi I.A.R., "Alternative parameter choices for multi-step quasi-Newton methods", *Optimization Methods and Software*, 2, June 1993, pp. 357-370.

[13] Ford, J.A., Narushima, Y. and Yabe, H., "Multi-step nonlinear conjugate gradient methods for unconstrained minimization", *Comput. Optim. Appl.*, 40, Nov. 2008, pp. 191-216.

[14] Gould, N.I.M., Orban, D. and Toint, Ph.L., CUTEr and SifDec: Aconstrained and unconstrained testing environment, revisited," ACM Transactions on Mathematical Software, 29, 2003, pp. 373-394.

[15] Hager W. and Zhang H.C., "A new conjugate gradient method with guaranteed descent and an efficient line search", *SIAM J.Optim.*, 16, Feb. 2005, pp.170–192.

[16] Hestenes, M.R. and Stiefel, E., "Methods of conjugate gradients for solving linear systems", *J. Res. Nat. Bur. Stan. Sec. B*, 48, July 1952, pp. 409–436.

[17] Liu, Y. and Storey, C., "Efficient generalized conjugate gradient algorithms, part1:theory", *J. Optim. Theory Appl*., 69, Feb. 1991, pp. 129–137.

[18] Moghrabi, I.A.R., "Numerical experience with multiple update quasi-newton methods for unconstrained optimization", *Journal of Mathematical Modeling and Algorithms*, 6, Jan. 2007, pp. 214-238.

[19] Moghrabi, I.A.R., "Implicit extra-update multi-step quasi-newton methods", Int. J. Operational Research, 28, August 2017, pp. 69-81.

[20] Moré, J.J., Garbow, B.S., Hillstrom, K.E., "Testing unconstrained optimization software", *ACM Trans. Math. Softw*., **7**, April 1981, pp. 17–41.

[21] Perry, A., "A modified conjugate gradient algorithm", *Oper. Res.* ,26, Jan. 1978, pp. 1073-1078.

[22] Polak, E. and Ribiére G., "Notesurla convergence de directions conjuguées", *Rev. Francaise Infomat Recherche Operatonelle*, 3e Année, 16, Dec. 1969, pp. 35–43.

[23] Polyak, B. T., "The conjugate gradient method in extreme problems", *USSR Comp. Math. Phys.*, 9, March 1969, pp. 94–112.

[24] Powell, M.J.D., "Restart procedures for the conjugate gradient method", *Math. Program.* , 12, Nov. 1977, pp. 241–254.

[25] Haelterman, R., Bogaers, A., Degroote, J. and Boutet, N., "Quasi-Newton Methods for the Acceleration of Multi-Physics Codes," IAENG International Journal of Applied Mathematics, vol. 47, no.3, pp 352-360, 2017.

[26] Salane, D. and Tewarson, R.P., "On Symmetric Minimum Norm Updates", *IMA Journal of Numerical Analysis*, 9, 1, Jan. 1983, pp. 235-240.

[27] Shanno, D.F., "Conditioning of quasi-Newton methods for function minimization", *Math. Comp.*, 24, May 1970, pp. 647-656.

[28] Shanno, D.F. and Phua, K.H., "Matrix conditioning and nonlinear optimization", *Math. Programming* ,14, May 1978, pp. 149-160.

[29] Shanno, D.F., "On the convergence of a new conjugate gradient algorithm", *SIAM J. Numer. Anal.*, 15, Jan. 1978, pp. 1247–1257.

[30] Zhang, W., Qiu, D. and Dong, M., "Optimizations of Convex and generalized Convex Fuzzy Mappings in The Quotient Space of Fuzzy Numbers", IAENG International Journal of Applied Mathematics, vol. 47, no. 4, pp 431-436, 2017.

[31] Wolfe, P., "Convergence conditions for ascent methods II: Some corrections", *SIAM Rev.*, 13, June 1971, pp. 185-188.

Issam A.R. Moghrabi is a Professor of M.I.S/C.S in the College of Business Administration at Gulf University for Science and Technology. He laid the foundations of the M.B.A Program at GUST while serving as the MBA Director for the past five years. He is a Fulbright Scholar and did a post-doctoral research in Geographic Information Systems. The author got his B.Sc. in Computer Science, with distinction, from the Lebanese American University, his M.Sc, with distinction, and Ph.D. from Essex

University. His main research interests are in mathematical Optimization, Management Science and Information Retrieval and database systems. This author became a Member (M) of IAENG in 2000. He is also a member of the Non-Linear Analyst Group and a referee for several reputable journals [email: Moughrabi.i@gust.edu.kw].