Conjugate Gradient Methods Using Quasi-Newton Updates with Inexact Line Searches

HANIF D. SHERALI

Department of Industrial Engineering and Operations Research,
Virginia Polytechnic Institute and State University,
Blacksburg, Virginia 24061-0118

AND

OSMAN ULULAR

AT & T Laboratories, Room 35B34, 2 Oak Way,
Berkeley Heights, NJ 07922

Submitted by Augustine O. Esogbue
Received February 17, 1987

Conjugate gradient methods are conjugate direction or gradient deflection methods which lie somewhere between the method of steepest descent and Newton's method. Their principal advantage is that they do not require the storage of any matrices as in Newton's method, or as in quasi-Newton methods, and they are designed to converge faster than the method of steepest descent. Unlike quasi-Newton or variable-metric methods, these are fixed-metric methods in which the search direction at each iteration is based on an approximation to the inverse Hessian constructed by updating a fixed, symmetric, positive definite matrix, typically the identity matrix. The resulting approximation is usually not symmetric, although some variants force symmetry and hence derive memoryless quasi-Newton methods. In this paper, we present a scaled modified version of the conjugate gradient method suggested by Perry, which employs the quasi-Newton condition rather than conjugacy under inexact line searches, in order to derive the search directions. The analysis is extended to the memoryless quasi-Newton modification of this method, as suggested by Shanno. Computational experience on standard test problems indicates that the proposed method, along with Beale and Powell's restarts, improves upon existing conjugate gradient strategies.

1. INTRODUCTION

Conjugate gradient methods offer a significant improvement over steepest descent algorithms at a modest increase in storage requirements, and are hence especially well suited to large-scale applications. They are conjugate direction algorithms which converge in at most $n$ iterations for
unconstrained quadratic optimization problems in $\mathbb{R}^n$ when using exact line searches. However, they are also applied to nonquadratic problems, because smooth functions exhibit quadratic behavior in the vicinity of the optimum. In such cases, the procedure is usually reset (reinitialized) every $n$ iterations in order to improve the rate of convergence. Various modifications of the conjugate gradient strategy have been suggested under the relaxation of the quadratic and/or the exact line search assumptions, and different restart conditions have been recommended.

The sequence of iterates for minimizing an unconstrained, differentiable function $f: \mathbb{R}^n \rightarrow \mathbb{R}$ by conjugate gradient methods is generated according to

$$x^{k+1} = x^k + \lambda_k d_k,$$

where

$$d_k = -g_k + \beta_k d_{k-1}. \tag{2}$$

Here, $x^k$, $g_k$, $d_k$, and $\lambda_k$ are respectively the iterate, the gradient $\nabla f(x^k)$ of $f(\cdot)$ at $x^k$, the search direction, and the step length to be taken along $d_k$ at iteration $k$. Most conjugate gradient methods differ in their choice for the multiplier $\beta_k$ used to construct the search direction given by (2). For example, Hestenes and Stiefel [7] derive $\beta_k$ by requiring the search direction $d_k$ to be $H_k$-conjugate to $d_{k-1}$, i.e., enforcing $d_k^T H_k d_{k-1} = 0$ (see Hestenes [6] or Luenberger [9]), where $H_k$ is the Hessian of $f(\cdot)$ at $x^k$. Note that using a quadratic approximation of $f(\cdot)$ at $x^k$ along with (1), we have

$$g_k - g_{k-1} = \lambda_{k-1} H_k d_{k-1}. \tag{3}$$

Hence, the conjugacy requirement gives

$$d_k^T (g_k - g_{k-1}) = 0. \tag{4}$$

Substituting (2) into (4) yields Hestenes and Stiefel's [7] choice (HS)

$$\beta_k = \frac{g_k^T (g_k - g_{k-1})}{d_{k-1}^T (g_k - g_{k-1})} = \frac{(g_k^T y_k) \lambda_{k-1}}{p_k^T y_k}, \tag{5}$$

where for convenience, we have denoted

$$y_k \equiv g_k - g_{k-1}$$

and

$$p_k \equiv (x^k - x_{k-1}) \equiv \lambda_{k-1} d_{k-1}. \tag{6}$$

The Polak and Ribiere [16] choice (PR) for $\beta_k$ is given by

$$\beta_k = g_k^T y_k / \|g_{k-1}\|^2, \tag{7}$$
CONJUGATE GRADIENT METHODS

and is a reduction of (5) when exact line searches are performed, since $g^T_k d_{k-1} = 0$ in this event. Furthermore, when $f(\cdot)$ is quadratic, all the directions generated via (2) using (5) or (7) are mutually conjugate, and the gradients of $f(\cdot)$ at the different iterates are mutually orthogonal. In particular, we have,

$$g^T_k g_{k-1} = 0.$$  

(8)

Hence, for quadratic functions and under exact line searches, the PR choice (7) further reduces to the Fletcher and Reeves [5] formula (FR)

$$\beta_k = \frac{\|g_k\|^2}{\|g_{k-1}\|^2}. \quad (9)$$

In fact, using exact line searches, although the (FR) and (PR) choices for $\beta_k$ are equivalent when $f(\cdot)$ is quadratic, the choice (PR) turns out to be preferable with nonquadratic functions as demonstrated by Powell [18]. However, in order to facilitate convergence to a point of zero gradient when using the PR choice, Powell [19] recommends a further modification in which $\beta_k$ is set to zero whenever (7) turns out to be negative. In the same spirit, when inexact line searches are performed, we would expect the HS formula (5) to be preferable, and this is confirmed by our computational results in Section 4. Note that when inexact line searches are performed, the conjugacy relationship holds only between consecutive directions, even when $f(\cdot)$ is quadratic. Nazareth [11] and Dixon et al. [3] propose alternate three-term recurrence relationships for generating mutually conjugate directions in this case, if so desired. However, we concentrate on directions generated via the traditional formula (2).

Perry [15] observes that using the (HS) choice, the direction $d_k$ in (2) can be rewritten as

$$d_k = - \left[ I - \frac{p_k y_k^T}{y_k^T p_k} \right] g_k = - Q_k^{HS} g_k. \quad (10)$$

Note that $Q_k^{HS}$ plays the role in (10) of an approximation to the inverse Hessian, but is not symmetric. Hence strictly speaking, it is not a memoryless quasi-Newton update (see Luenberger [9]). More pertinently, if $Q_k$ is denoted as some approximation to the inverse Hessian, the quasi-Newton condition requires that $Q_k y_k = p_k$, or under symmetry, that

$$y_k^T Q_k = p_k^T. \quad (11)$$

Aside from nonsymmetry, (10) yields $y_k^T Q_k^{HS} = 0$. Now, Perry [15] notes that under inexact line searches, it is more appropriate to choose $d_k$ as $- Q_k^P g_k$, where ($P$) denotes Perry's choice and the inverse Hessian
approximation \( Q_k^p \) is chosen to satisfy the (quasi-Newton) condition (11) rather than simply \( H_k \)-conjugacy. Hence, \( v_k' d_k = y_k' Q_k^p g_k = -p_k' g_k \) must hold. Substituting this in (2) gives Perry's choice for \( \beta_k \) and the corresponding direction \( d_k \) as

\[
\beta_k = \frac{y_k' g_k - p_k' g_k}{y_k' d_k} \quad (12)
\]

\[
d_k = - \left[ I - \frac{p_k y_k'}{y_k' p_k} + \frac{p_k p_k'}{y_k' p_k} \right] g_k \equiv -Q_k^p g_k. \quad (13)
\]

Note that an extra term is added to \( Q_k^H \) to yield the approximation \( Q_k^p \), and that the choice (P) given by (12) is identical to (HS)/(PR) if exact line searches are performed. However, Shanno [21] observes \( Q_k^p \) is not symmetric and so, although (11) holds, the true quasi-Newton condition \( Q_k y_k = p_k \) is not satisfied with \( e_k = Q_k^p \). By adding the term \( -y_k p_k' / y_k' p_k \) in (13) to make the new \( Q_k^p \) symmetric, and then forcing the quasi-Newton condition to hold, Shanno [21] derives the search direction

\[
d_k = -Q_k^{MBFGS} g_k,
\]

where

\[
Q_k^{MBFGS} = I - \frac{(p_k y_k' + y_k p_k')}{y_k' p_k} + \left[ 1 + \frac{y_k y_k'}{y_k' p_k} \right] \frac{p_k p_k'}{y_k' p_k}. \quad (14)
\]

As observed by Shanno [21], this corresponds to a memoryless BFGS update, and hence the notation MBFGS in (14). Again, with exact line searches, this reduces to the (HS)/(PR) strategy. Additionally, using Beale’s [2] restart criterion triggered by a condition due to Powell [18] (see Section 3 below), Shanno [21] proposed the following alternative algorithm. Let \( t \) denote the iteration at which a restart is performed under Powell’s [18] criterion. Shanno first performs a memoryless update on the identity matrix using Oren and Spedicato’s [14] self-scaling conjugate gradient strategy at the restart iteration \( t \), and then updates this at the current iteration \( k \) via the standard BFGS update scheme (see, e.g., Luenberger [9, Chap. 9]). This leads to the search direction which is reproduced for convenience below to correct typographical errors in Eqs. (34) and (38) in [21, p. 250]:

\[
d_k = -\hat{H}_k g_k + \frac{p_k g_k}{p_k' y_k} \hat{H}_k y_k - \left[ \left( 1 + \frac{y_k' \hat{H}_k y_k}{p_k' y_k} \right) \frac{p_k g_k}{p_k' y_k} - \frac{y_k' \hat{H}_k g_k}{p_k' y_k} \right] p_k, \quad (15)
\]
where

\[ \hat{H}_k g_k = \frac{p^*_k y^*_\tau}{y^*_\tau y^*_\tau} g_k - \frac{p^*_k g_k}{y^*_\tau y^*_\tau} y^*_\tau + \left( 2 \frac{p^*_k g_k}{p^*_k y^*_\tau} - \frac{y^*_k g_k}{y^*_\tau y^*_\tau} \right) p^*_\tau \]  

(16a)

\[ \hat{H}_k y_k = \frac{p^*_k y^*_\tau}{y^*_\tau y^*_\tau} y_k - \frac{p^*_k y_k}{y^*_\tau y^*_\tau} y^*_\tau + \left( 2 \frac{p^*_k y_k}{p^*_k y^*_\tau} - \frac{y^*_k y_k}{y^*_\tau y^*_\tau} \right) p^*_\tau. \]  

(16b)

Hence, the above procedure due to Shanno [21] employs a sequential double quasi-Newton update, and is therefore abbreviated as (SDQN).

The foregoing strategy is motivated by the superior convergence rate properties of quasi-Newton methods over traditional conjugate gradient methods (see McCormick and Ritter [10]). Note that as shown by Nazareth [12], when minimizing a quadratic function with a positive definite Hessian and employing exact line searches, the conjugate gradient method and the BFGS quasi-Newton method generate identical search directions at each iteration when initialized at the same point. However, for use with nonquadratic functions, Nazareth [12, 13] proposes combinations of conjugate gradient algorithms with quasi-Newton methods, which either employ less frequent updates or employ updates over reduced subspaces defined at each iteration by using some recent gradients and/or directions. Observe that Shanno's [21] SDQN method described above is a member of this class of algorithms.

In the sequel, we extend Perry's [15] approach to yield a scaled version of (12)–(13), and of the memoryless quasi-Newton update scheme (14). Different strategies attempted for selecting the scaling parameter are discussed. One particularly simple choice, however, produces consistently superior results, and is therefore chosen for implementation. All of the foregoing existing methods and the proposed algorithms are tested using Beale's [2] restart scheme along with Powell's [18] restart condition, and using an inexact quadratic interpolation line search technique. The computational results establish the relative advantage of the proposed modifications over the previous procedures.

2. PROPOSED STRATEGIES BASED ON SCALED QUASI-NEWTON UPDATES

We now assume throughout that line searches are performed inexactly. Following Perry [15], our scheme relies on the quasi-Newton condition (11), rather than conjugacy, in order to derive the multiplier \( \beta_k \) in (2). The method is motivated by the fact that Perry's derivation for \( \beta_k \) in (12) is essentially based on equating \( d_k = -g_k + \beta_k d_{k-1} \) to \( -Q_k g_k \), where \( Q_k \) is an approximation to the inverse Hessian, albeit nonsymmetric. However,
suppose that it turns out that the Newton direction \(-H_k^{-1}g_k\) is in fact contained in the cone spanned by \(-g_k\) and \(d_{k-1}\), and not coincident with \(d_{k-1}\). Then \(\beta_k\) cannot alone ensure equality of \(d_k \equiv -g_k + \beta_k d_{k-1}\) and \(-H_k^{-1}g_k\). This is because, under these assumptions, \(d_k\) and the Newton direction are only guaranteed to be collinear. However, by using an appropriate scale parameter \(s_k\), we can write

\[ s_k d_k \equiv s_k [-g_k + \beta_k d_{k-1}] = -H_k^{-1}g_k. \tag{17} \]

Transposing both sides and multiplying with \(H_k d_{k-1}\) yields

\[ s_k [-g_k + \beta_k d_{k-1}]^T H_k d_{k-1} = -g_k^T d_{k-1}. \tag{18} \]

But from (3), \(H_k d_{k-1} = y_k/\lambda_{k-1}\). Using this along with (6) in (18) and solving for \(\beta_k\), we obtain

\[ \beta_k = \frac{y_k^T g_k - (1/s_k) p_k^T g_k}{y_k^T d_{k-1}}. \tag{19} \]

We abbreviate this modified choice of \(\beta_k\) as \((P-SU)\) for convenience. Notice from (12) that with \(s_k = 1\), this coincides with Perry's strategy \((P)\), and as \(s_k \to \infty\), this coincides with the \((HS)\) choice \((5)\). Hence, \((19)\) provides a unifying framework, yielding a continuous variation between Hestenes and Stiefel's \([7]\) and Perry's \([15]\) strategies. Note that with exact line searches, \(p_k^T g_k = 0\), and therefore the choice of \(s_k\) is immaterial, and convergence for quadratic functions is obtained in at most \(n\) iterations as before. Also, observe that when inexact line searches are performed, the quantity \(-p_k^T g_k\) is likely to be positive if the step length \(\lambda_{k-1}\) is too short, and negative if it is too long relative to the optimal step length. Hence, when \(s_k\) lies in the range \((0, 1)\) and approaches zero, assuming that the denominator of \(\beta_k\) in \((19)\) is positive (which is likely if the line search is not too inaccurate), the quantity \(\beta_k\) will become large in magnitude and positive in the former case, and negative in the latter case. Consequently from (2), the search direction \(d_k\) at \(x^k\) will be aligned closer to \(d_{k-1}\) in the first case, and to \(-d_{k-1}\) in the second case, thus forcing a more accurate line search.

Next, we need to specify a suitable value for the scaling parameter \(s_k\). Note that in the above scenario, since a unit step length along the Newton direction is optimal with respect to the quadratic approximation, \(s_k\) in \((17)\) has the interpretation of being an optimal step length that needs to be taken along \(d_k = -g_k + \beta_k d_{k-1}\) using this approximation of \(f(\cdot)\). However, \(\beta_k\) is itself a function of \(s_k\) in \((19)\). Hence, this leads to the following derivation. First, using a quadratic approximation \(f(x^k + \lambda d_k) = f(x^k) + \lambda d_k^T g_k + \frac{1}{2} \lambda^2 d_k^T H_k d_k\) for the line search, the optimal step length to
be taken along a direction $d_k$ at $x^k$ is given by $\lambda^* = -d^*_k g_k / d^*_k H_k d_k$. Equating this $\lambda^*$ to $s_k$ in view of the above interpretation, we get, using (2),

$$s_k = -\frac{d^*_k g_k}{d^*_k H_k d_k} = \frac{\|g_k\|^2 - \beta_k g_k^* d_{k-1}^*}{(-g_k + \beta_k d_{k-1})^t H_k (-g_k + \beta_k d_{k-1})}.$$  \hspace{1cm} (20)

Let us denote the denominator of (20) as $D$. Using (3), we get that $D \equiv g^*_k H_k g_k + \beta_k (\beta_k d_{k-1}^* - 2g_k^* d_k) H_k d_k$, is given by

$$D = g^*_k H_k g_k + (\beta_k / \lambda_{k-1}) [\beta_k d_{k-1}^* y_k - 2g_k^* y_k].$$ \hspace{1cm} (21)

Now, we can use a second-order Taylor's expansion of $f(\cdot)$ at $x^k$ to approximately write

$$g^*_k H_k g_k = (2/\alpha^2) [f(x^k + \alpha g_k) - f(x^k) - \alpha \|g_k\|^2],$$ \hspace{1cm} (22)

where $\alpha$ is some small positive constant, say, $\alpha = 0.01$. Substituting (21) and (22) into (20), and using formula (19) for $\beta_k$, we obtain a linear equation in $s_k$ which yields

$$s_k = \frac{\|g_k\|^2 - (g_k^* d_{k-1}) (g_k^* y_k) / (d_{k-1}^* y_k)}{(2/\alpha^2) [f(x^k + \alpha g_k) - f(x^k) - \alpha \|g_k\|^2] - (g_k^* y_k)^2 / \lambda_{k-1} d_{k-1}^* y_k}.$$ \hspace{1cm} (23)

Observe that $s_k$ in (23) need not be positive in general. Whenever $s_k \leq 0$ in (23), one can resort to selecting $s_k$ as the actual previous step length $\lambda_{k-1}$, or simply use $s_k = 1$ (Perry's choice), or take $s_k \to \infty$ (Hesteness and Stiefel's choice). Since a nonpositive value of $s_k$ does not appear to favor the use of the quasi-Newton type of condition (11), we opted for the lattermost strategy whenever $s_k \leq 0$. Hence, we first compute

$$s_k \text{ given by (23) in case this is positive, and } s_k \to \infty \text{ otherwise.} \hspace{1cm} (C1)$$

Next, we substitute this value of $s_k$ in (19) to obtain $\beta_k$, and finally, use this $\beta_k$ in (2) in order to obtain the search direction $d_k$. Note that for quadratic functions and using exact line searches, convergence in at most $n$ iterations is obtained as before. Furthermore, in this case, $s_k$ in (23) is itself the optimal step length to be taken, and hence, no line search needs to be performed. For the nonquadratic case, since $s_k$ given by (23) is based on a quadratic approximation for $f(\cdot)$, it remains a viable approximate step length. However, its use gave mixed computational results, and so we chose not to employ this prescribed step length.

As an alternative to the choice (C1), one can recommend the actual previous step length $\lambda_{k-1}$ to be used as an estimator for $s_k$. This yields...
another particularly simple choice (C2), which does not involve the additional functional evaluations and computations in (23):

\[ s_k = \lambda_{k-1}. \quad \text{(C2)} \]

There are several other choices of \( s_k \) which we attempted, including the smoothing option \( s_k = (s_{k-1} + \lambda_{k-1})/2 \) with \( s_1 \equiv 1 \), and the option of selecting \( s_k \) as a least-squares minimizer of \( \|s_k[H_k g_k - \beta_k H_k d_{k-1}] - g_k\|^2 \) in view of (17). However, the above choices (C1) and (C2) performed consistently better, and hence, we omit further details of these alternatives.

**Illustrative Example**

To illustrate, consider the unconstrained minimization of the quadratic objective function \( f(x) = 4x_1^2 + 4x_2^2 - 4x_1x_2 - 12x_2 \) defined on \( \mathbb{R}^2 \), taken from Bazaraa and Shetty [11]. Arbitrarily, let us assume that \( x_{k-1} = (-4, 1)' \), that the direction of motion at \( x_{k-1} \) is \( d_{k-1} = (1, 0)' \), and that we have taken an inexact step length of \( \alpha_{k-1} = 4 \) at \( x_{k-1} \) along \( d_{k-1} \), in order to arrive at \( x_k = (0, 1)' \). (The exact step length is unity, and \( k \) may be equal to 2.) From (6), let us also compute

\[ g_{k-1} = (-8, -2)', \quad g_k = (-4, -4)', \quad y_k = (4, -2)' \quad \text{and} \quad p_k = (\frac{1}{2}, 0)' \quad \text{(24)} \]

Now, let us compute \( d_k \) in (2) using (i) the conjugacy relationship which leads to the (HS) choice (5), (ii) using Perry’s [15] formula (12), and (iii) using our strategy with choice (C1) above.

**Case (i).** The \( H \)-conjugacy relationship yields via (5) that \( \beta_k = -2 \). Since the step length \( \lambda_{k-1} \) was short of the exact value, the fact that \( \beta_k < 0 \) in obtaining \( d_k \) as \( H \)-conjugate to \( d_{k-1} \) is to be expected. From (2), this gives the direction \( d_k = (2, 4)' \). The optimal step length along this direction is \( \frac{1}{4} \), leading to the point \( (\frac{1}{2}, 2)' \), which is not optimal. Of course, the conjugate direction \( (2, 4)' \), would have led to the optimal solution if an exact line search had been performed at \( x^k \).

**Case (ii).** Using Perry’s formula (P) in (12) gives \( \beta_k = -\frac{3}{2} \). This leads to \( d_k = (\frac{3}{2}, 4)' \) via (2). Note that the Newton direction at \( x^k \) is \( -H^{-1}g_k \equiv (1, 1)' \) so that a unit step length along this direction would lead to the optimal solution \( (1, 2)' \). However, the direction \( d_k = (\frac{3}{2}, 4)' \) has an associated optimal step length of \( \sqrt{\frac{5}{8}} \), leading to the iterate \( (\frac{5}{8}, \frac{10}{3})' \).

**Case (iii).** Recognizing the scale factor \( s_k \) which must be used in (17) along with the quasi-Newton condition in order to recover the Newton direction if possible, we obtain via (23) that \( s_k = \frac{1}{4} \). Substituting this into (19) yields \( \beta_k = 0 \), and leads to \( d_k = (4, 4)' \) via (2), which is \( 1/s_k = 4 \) times the Newton direction \( (1, 1)' \) at \( x^k \). Now, minimizing \( f(\cdot) \) from \( x^k = (0, 1)' \)
along $d_k = (4, 4)'$ requires an optimal step length of $\frac{1}{4}$, which is indeed $s_k$ itself, yielding the optimal solution $(1, 2)'$. Hence, this illustrates the advantage of the scaling modification of Perry's [15] algorithm.

**Derivation of a Scaled Version of the MBFGS Update (14)**

Recall that as pointed out by Shanno [21], Perry's [15] method when made symmetric and forced to satisfy the quasi-Newton condition yields the memoryless BFGS update in (14). In the same spirit, when $\beta_k$ is given by (19) via our strategy above, we obtain from (2) that

$$d_k = -Q_k^{P, SU} g_k, \quad \text{where} \quad Q_k^{P, SU} = \left[ I - \frac{p_k y_k}{y_k' p_k} + \frac{p_k p_k'}{s_k y_k' p_k} \right].$$

Hence, following Shanno [21], suppose that we make $Q_k^{P, SU}$ symmetric by adding to it the term $(-y_k p_k' / y_k' p_k)$, and thereby obtain the update

$$\bar{Q}_k = I - \frac{p_k y_k'}{y_k' p_k} - \frac{y_k p_k'}{y_k' p_k} + \frac{p_k p_k'}{s_k y_k' p_k}.$$  \hspace{1cm} (26)

Now, from (17), since $s_k d_k$ is supposed to be our quasi-Newton direction, where $d_k = -Q_k g_k$ for some update $Q_k$, we want $s_k Q_k$ to satisfy the quasi-Newton condition $s_k Q_k y_k = p_k$. However, from (26), we have

$$s_k \bar{Q}_k y_k = P_k - s_k P_k (y_k' y_k / y_k' p_k).$$

Therefore, in order to force the quasi-Newton condition while maintaining the symmetry in $\bar{Q}_k$, we add the term $(y_k' y_k / y_k' p_k) [p_k p_k' / y_k' p_k]$ to $\bar{Q}_k$. This gives the required $Q_k \equiv Q_k^{MBFGS-SU}$, say, which is our modification of $Q_k^{MBFGS}$ in (14). (It is easily verified that the (scaled) quasi-Newton condition $s_k Q_k^{MBFGS-SU} y_k = p_k$ holds.) We hence obtain

$$d_k = -Q_k^{MBFGS-SU} g_k,$$  \hspace{1cm} (28a)

where

$$Q_k^{MBFGS-SU} = I - \frac{[p_k y_k' + y_k p_k']}{y_k' p_k} + \left[ \frac{1}{s_k} + \frac{y_k' y_k}{y_k' p_k} \right] p_k p_k'. $$

Notice that when $s_k = 1$, (28) coincides with the memoryless BFGS update (14). Also, whenever exact line searches are performed, $p_k' g_k$ is zero and so $d_k$ in (28) reduces to the Hestenes and Stiefel [7] direction given by (2) and (5). Hence, the conjugacy relationship holds in this case and so convergence is obtained in at most $n$ iterations for quadratic functions. Again, a derivation similar to (23) may be used to estimate $s_k$, or we may simply use $s_k \equiv \lambda_{k-1}$ as in (C2).
3. INCORPORATING BEALE AND POWELL’S RESTART CRITERIA

It is well known that the rate of convergence of a conjugate gradient algorithm is only linear unless if it is appropriately restarted occasionally, at least every \( n \) iterations. (See McCormick and Ritter [10] and Hestenes [6].) Instead of restarting at some iteration \( \tau \) with the negative gradient direction, Beale [2] suggested that one ought to restart by using the computed direction \( d \), itself, which contains useful second-order type of information. At the next iteration \( k = \tau + 1 \), the direction \( d_k \) is computed as
\[
-g_k + \beta_k d_{k-1},
\]
i.e., \( d_{\tau+1} = -g_{\tau+1} + \beta_{\tau+1} d_{\tau} \), where \( \beta_k \) is derived using any appropriate formula. However, in order to maintain conjugacy for iterations \( k > \tau + 1 \), Beale [2] shows that one needs to use an additional term in (2) as given below:
\[
d_k = -g_k + \beta_k d_{k-1} + \gamma_k d_{\tau}, \tag{29a}
\]
where
\[
\gamma_k = g_k' v_{\tau+1} / d_k' v_{\tau+1}. \tag{29b}
\]

Powell [18] suggested that one should use Beale’s restart technique by setting \( \tau = k - 1 \) either whenever
\[
|g'_{k-1} g_k| \geq 0.2 \|g_k\|^2 \tag{30}
\]
holds or every \( n \) iterations, whichever occurs first. Additionally, Powell suggested that one ought to check if the search direction is a sufficiently descent direction by verifying if
\[
-1.2 \|g_k\|^2 \leq d_k' g_k \leq -0.8 \|g_k\|^2 \tag{31}
\]
holds. In case (31) is not satisfied, then Powell recommends that the procedure may be reset by putting \( \tau = k - 1 \), and redefining \( d_k \) via (29) using \( \gamma_{\tau+1} \equiv 0 \).

This strategy is shown to yield very favorable results when used with the (HS) choice (5), as compared with the standard (PR) and (FR) approaches. Also, Shanno [21] reports that this Beale–Powell restart criterion leads to a significant improvement in the overall convergence properties of his proposed algorithms. Our experience also supports this result, and hence this restart criterion is used with all the strategies tested herein. Additionally, observe that if exact line searches are used, then when the procedure is reset at \( \tau = k - 1 \), the direction \( d_k \) at \( k = \tau + 1 \) is a descent direction, since \( d_k' g_k = -\|g_k\|^2 \) in this case. However, with inexact line searches, if this \( d_k \) is not a descent direction, we reset with \( \tau = k \) and use
$d_k = -g_k$ as in Perry [15]. For a modified three-term conjugate gradient scheme with alternative restart procedures when inexact line searches are performed, we refer the reader to Dixon et al. [3] and Nazareth [11].

4. Computational Results

Methods Tested

In this section, we provide computational experience with the methods of Hestenes and Stiefel (HS) [7], Polak and Ribiere (PR) [16], Fletcher and Reeves (FR) [5], and Perry (P) [15], as well as with Shanno [21] and Perry's [15] memoryless BFGS update strategy (MBFGS), Shanno's [21] sequential double quasi-Newton strategy (SDQN), and our proposed modifications (P-SU) of Perry's strategy and (MBFGS-SU) of the memoryless BFGS strategy of Section 2. These methods are given respectively by Eqs. (5), (7), (9), (12), (14), (15, 16), (19), and (28), used in conjunction with (2) where appropriate. Additionally, we report experience using (C1) and (C2) in (19) with (P-SU), and similarly with (MBFGS-SU). The Beale-Powell restart criterion embodied by (29), (30), and (31) was used with all the methods in the same manner, except that Shanno's [21] algorithm (SDQN) has this restart built into it in (15, 16). Note that unlike Shanno [21], we add the term $y_k d_t$ in (29) to the MBFGS and the MBFGS-SU strategies as well, so that with quadratic functions and exact line searches, convergence is obtained in at most $n$ steps. We found that this significantly improves both the latter strategies.

Test Problems

Standard test problems, along with recommended starting solutions, available from various sources in the literature were used for the comparisons. These problems include those used by Perry [15] and by Shanno [21]. For convenience, these problems are reproduced in the Appendix, and are indexed in this same order along with the values of $n$ in the tables below.

Overall Convergence Criterion

All the methods were assumed to have converged when each component of the gradient vector was less than $10^{-5}$. As mentioned by Shanno [21], although this is not an ideal criterion and it is typically more difficult to achieve than those based on objective function values, it is more realistic since the optimal objective value is usually unknown. Whenever this criterion was not satisfied within 400 iterations, the method was said to have failed in such an instance. This is indicated by an $F$ in the tables below.
Two types of standard quadratic interpolation line searches were attempted for the problem of minimizing \( \phi(\lambda) = f(x^k + \lambda d_k), \ \lambda \geq 0 \) (see Luenberger [9]). These methods are designated LS1 and LS2 below. In LS1, three points \( \lambda_1 < \lambda_2 < \lambda_3 \) are first determined such that \( \phi(\lambda_1) \geq \phi(\lambda_2) \leq \phi(\lambda_3) \), with at least one inequality strict. The value of \( \lambda_1 \) is taken as zero. If \( \phi(\lambda_{k-1}) < \phi(0) \), then we set \( \lambda_2 = \lambda_{k-1} \), and otherwise, we set \( \lambda_3 = \lambda_{k-1} \). In the former case, \( \lambda_3 \) is found by doubling \( \lambda_1 \) sequentially until the required condition on the three points is satisfied, and in the latter case, the value of \( \lambda_2 \) is found by halving \( \lambda_3 \) sequentially until this condition is satisfied. A quadratic curve is constructed through these points, and its closed-form minimizer \( \lambda \) is determined. The value of \( \lambda_{k} \) is then chosen as \( \text{argmin} \{\phi(\lambda): \lambda \text{ equals } \lambda_2 \text{ or } \lambda_3\} \).

For the line search LS2, after \( \lambda_k = \lambda^*_k \), say, is determined as in LS1 above, if the quadratic fit minimizer \( \lambda = \lambda_2 \) then the procedure halts with \( \lambda \) as the prescribed step length. Otherwise, using \( \lambda \) and two of the three points \( \lambda_1, \lambda_2, \) and \( \lambda_3 \), a set of three points satisfying the condition stated in LS1 is obtained as follows. If \( \lambda > \lambda_2 \), then these three points are selected as \( (\lambda_2, \lambda, \lambda_3) \) if \( \phi(\lambda) \leq \phi(\lambda_2) \), and as \( (\lambda_1, \lambda_2, \lambda_3) \) if \( \phi(\lambda) > \phi(\lambda_2) \). On the other hand, if \( \lambda < \lambda_2 \), then these three points are selected as \( (\lambda_1, \lambda, \lambda_2) \) if \( \phi(\lambda) \leq \phi(\lambda_2) \), and as \( (\lambda, \lambda_2, \lambda_3) \) if \( \phi(\lambda) > \phi(\lambda_2) \). (Note that if \( \phi(\lambda) = \phi(\lambda_2) \), then the third point selected above from \( \lambda_1 \) or \( \lambda_3 \) should have a larger \( \phi(\cdot) \) value.) Then, a quadratic fit is constructed for these three points and a revised solution \( \lambda^*_k \) is determined as in LS1 above. This process is continued until for some \( \lambda_k = \lambda^*_k, q \geq 1 \), any one of the following conditions holds: \( |f(x^k + \lambda^*_k d_k) - f(x^k)| \leq 10^{-5} |f(x^k)| \), or \( |\lambda^*_k - \lambda^*_{k-1}| \leq 10^{-5} \) (where \( \lambda^*_k \) is taken as the initial value of \( \lambda_2 \) in LS1), or \( |d_{k-1} g_k| < c |d_{k-1} g_k| \), where \( d_{k+1} = \nabla f(x^k + \lambda^*_k d_k) \). A value of \( c = 0.1 \) as recommended by Shanno [21] was attempted. (We also tried \( c = 0.9 \), but this gave results almost identical to the line search LS1.) We remark here that the foregoing line search techniques are practical, empirically attractive schemes. However, for ensuring theoretical convergence, one would need to operate LS2, for example, until some inexact line search termination criterion such as that of Armijo, or Goldstein, or Wolfe is satisfied (see Luenberger [9]).

Statistics Reported

For each problem solved, we report the total number of (major) iterations (ITER), and the total number of function and gradient calls (IFN) used by each procedure. (The latter quantity is given in brackets for ease in reading.) An \( F \) indicates that the convergence criterion was not met in 400 iterations, as noted above. Also, for each problem solved, we ranked the eight methods using average ranks, in order of increasing values of
Methods which had IFN values within 10% of each other were considered as ties, and were given common average ranks. Hence, for each problem the ranks sum to $1 + 2 + \cdots + 8 = 36$. When several values of $n$ were tried for any test problem, the ranks over these problems were averaged for each method. Finally, the ranks over all the problems were summed for each method, and these sums were then normalized by dividing through by the smallest such sum. These quantities are respectively denoted as the Total Ranks and the Total Normalized Ranks.

**Results**

We first attempted to study the performance of the choices (Cl) versus (C2) for selecting the scaling parameter $s_k$ to be used in (19) and (28) for the proposed strategies (P-SU) and (MBFGS-SU), respectively. (A derivation similar to (23) was used for (Cl) in connection with (MBFGS-SU).) Table I gives the results using the line search LS1. Note that with the

<table>
<thead>
<tr>
<th>Problem</th>
<th>(P-SU) ITER(IFN)</th>
<th>(P-SU) ITER(IFN)</th>
<th>MBFGS-SU ITER(IFN)</th>
<th>MBFGS-SU ITER(IFN)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 (n = 2)</td>
<td>4(41)</td>
<td>5(28)</td>
<td>4(41)</td>
<td>6(41)</td>
</tr>
<tr>
<td>2 (n = 2)</td>
<td>24(174)</td>
<td>24(164)</td>
<td>24(165)</td>
<td>23(169)</td>
</tr>
<tr>
<td>3 (n = 4)</td>
<td>65(427)</td>
<td>64(352)</td>
<td>76(504)</td>
<td>58(330)</td>
</tr>
<tr>
<td>4 (n = 4)</td>
<td>22(178)</td>
<td>20(142)</td>
<td>32(243)</td>
<td>35(251)</td>
</tr>
<tr>
<td>5 (n = 2)</td>
<td>8(56)</td>
<td>8(48)</td>
<td>8(61)</td>
<td>8(46)</td>
</tr>
<tr>
<td>6 (n = 5)</td>
<td>40(227)</td>
<td>42(201)</td>
<td>37(208)</td>
<td>43(205)</td>
</tr>
<tr>
<td>7 (n = 10)</td>
<td>85(464)</td>
<td>80(353)</td>
<td>71(386)</td>
<td>76(343)</td>
</tr>
<tr>
<td>8 (n = 20)</td>
<td>129(688)</td>
<td>124(535)</td>
<td>136(730)</td>
<td>146(639)</td>
</tr>
<tr>
<td>9 (n = 20)</td>
<td>259(1360)</td>
<td>273(1163)</td>
<td>262(1371)</td>
<td>261(1118)</td>
</tr>
<tr>
<td>10 (n = 20)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table I: Comparison of Strategies (C1) and (C2) for Selecting the Scaling Parameter
method (P-SU), the strategy (C1) takes at least 10% more gradient and function evaluations than does (C2) for all the test problems, except for Problems 8 and 9 (with $n = 10$), where it performs about 10% better than (C2). Similarly, with the method (MBFGS-SU), the strategy (C1) performs about the same or worse than the strategy (C2). Hence, we recommend the strategy (C2), and all the results given below use this choice for selecting $s_k$. However, for less nonquadratic problems, one may expect strategy (C1) to be more competitive.

Table II gives the results using the line search LSl. The proposed scaling modifications (P-SU) and (MBFGS-SU) both appear to provide an improvement over (P) and (MBFGS), respectively. In particular, the proposed strategy (P-SU) appears to dominate the other strategies tested. The overall relative performance of the methods is evidenced by the Total Ranks and the Total Normalized Ranks given in Table II.

Table III gives the results using the line search strategy LS2. Observe that on an overall comparison of Tables II and III, the line search strategy LSl seems to be preferable. In particular, for the proposed method (P-SU), the line search strategy LS2 performs better on 4 problems, worse on 5 problems, and about the same on the remaining 11 problems as compared with LSl. Again, comparing the relative performances of the different methods, the proposed method (P-SU) dominates all the other methods. This dominance seems to be more pronounced here than under the line search strategy LSl in Table II. This perhaps indicates that the proposed modifications generate better search directions which enhance their relative performance when using more accurate line searches.

Finally, we mention that we also attempted to run the algorithms by restarting with the negative gradient every $n$ iterations. Similar to Powell [18] and Shanno [21], we obtained significantly worse results than with the use of the Beale–Powell restart criterion. On a relative basis, the proposed modification (MBFGS-SU) appeared to perform best in this case. On a Total Rank basis, this algorithm gave a Total Rank value of 35.0, while the methods (HS), (P-SU), (PR), (MBFGS), (P), and (FR) gave Total Rank values of 38, 38.5, 39, 39, 45, and 45.5, respectively.

In conclusion, we have suggested in this paper a scaled version of Perry's method and of the memoryless quasi-Newton algorithm. Both these methods, like their predecessors, are derived through the use of the quasi-Newton condition rather than the conjugacy relationship, under inexact line searches. However, these methods further recognize that there is a scale factor which must be attached to the conjugate gradient direction when it is used as an estimator for the Newton direction. Using Beale and Powell's restart criterion, the scaled version of Perry's method, where the scale parameter is chosen simply as the previous step length, is shown to yield a promising conjugate gradient method.
<table>
<thead>
<tr>
<th>Problem</th>
<th>(FR) ITER(IFN)</th>
<th>(IS) ITER(IFN)</th>
<th>(PR) ITER(IFN)</th>
<th>(P) ITER(IFN)</th>
<th>(P-SU) ITER(IFN)</th>
<th>MBFGS ITER(IFN)</th>
<th>MBFGS-SU ITER(IFN)</th>
<th>SDQN ITER(IFN)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 (n=2)</td>
<td>7(44)</td>
<td>5(38)</td>
<td>5(29)</td>
<td>5(34)</td>
<td>5(28)</td>
<td>5(34)</td>
<td>6(41)</td>
<td>4(28)</td>
</tr>
<tr>
<td>2 (n=2)</td>
<td>32(187)</td>
<td>25(135)</td>
<td>23(158)</td>
<td>26(175)</td>
<td>24(164)</td>
<td>24(143)</td>
<td>23(169)</td>
<td>25(207)</td>
</tr>
<tr>
<td>3 (n=4)</td>
<td>F</td>
<td>72(410)</td>
<td>82(480)</td>
<td>71(405)</td>
<td>64(352)</td>
<td>65(421)</td>
<td>58(330)</td>
<td>65(454)</td>
</tr>
<tr>
<td>4 (n=4)</td>
<td>98(580)</td>
<td>36(256)</td>
<td>22(142)</td>
<td>27(218)</td>
<td>20(142)</td>
<td>42(278)</td>
<td>35(251)</td>
<td>34(284)</td>
</tr>
<tr>
<td>5 (n=2)</td>
<td>19(103)</td>
<td>8(43)</td>
<td>10(62)</td>
<td>8(44)</td>
<td>8(48)</td>
<td>8(50)</td>
<td>8(46)</td>
<td>8(56)</td>
</tr>
<tr>
<td>6 (n=5)</td>
<td>375(1510)</td>
<td>47(229)</td>
<td>42(206)</td>
<td>46(222)</td>
<td>42(201)</td>
<td>54(282)</td>
<td>43(205)</td>
<td>40(209)</td>
</tr>
<tr>
<td>7 (n=10)</td>
<td>F</td>
<td>74(338)</td>
<td>94(427)</td>
<td>79(366)</td>
<td>80(353)</td>
<td>78(347)</td>
<td>76(343)</td>
<td>74(343)</td>
</tr>
<tr>
<td>8 (n=20)</td>
<td>135(577)</td>
<td>157(667)</td>
<td>138(593)</td>
<td>124(535)</td>
<td>143(610)</td>
<td>146(639)</td>
<td>124(552)</td>
<td></td>
</tr>
<tr>
<td>9 (n=50)</td>
<td>F</td>
<td>265(1114)</td>
<td>337(1429)</td>
<td>273(1155)</td>
<td>273(1163)</td>
<td>270(1133)</td>
<td>261(1118)</td>
<td>278(1209)</td>
</tr>
<tr>
<td>10 (n=20)</td>
<td>21(113)</td>
<td>16(97)</td>
<td>16(96)</td>
<td>18(107)</td>
<td>15(92)</td>
<td>16(94)</td>
<td>15(91)</td>
<td>15(96)</td>
</tr>
<tr>
<td>11 (n=40)</td>
<td>31(157)</td>
<td>22(125)</td>
<td>23(128)</td>
<td>25(147)</td>
<td>21(120)</td>
<td>23(129)</td>
<td>21(120)</td>
<td>12(125)</td>
</tr>
<tr>
<td>12 (n=100)</td>
<td>50(239)</td>
<td>34(179)</td>
<td>34(177)</td>
<td>35(181)</td>
<td>33(170)</td>
<td>36(190)</td>
<td>34(175)</td>
<td>34(194)</td>
</tr>
<tr>
<td>13 (n=200)</td>
<td>72(327)</td>
<td>48(227)</td>
<td>48(230)</td>
<td>49(231)</td>
<td>47(225)</td>
<td>51(248)</td>
<td>48(227)</td>
<td>49(256)</td>
</tr>
<tr>
<td>14 (n=3)</td>
<td>134(577)</td>
<td>25(147)</td>
<td>21(155)</td>
<td>25(146)</td>
<td>27(150)</td>
<td>25(141)</td>
<td>22(132)</td>
<td>24(151)</td>
</tr>
<tr>
<td>15 (n=5)</td>
<td>208(989)</td>
<td>45(263)</td>
<td>44(242)</td>
<td>36(216)</td>
<td>51(298)</td>
<td>36(222)</td>
<td>42(260)</td>
<td>40(286)</td>
</tr>
<tr>
<td>16 (n=10)</td>
<td>F</td>
<td>169(1208)</td>
<td>228(1639)</td>
<td>314(2363)</td>
<td>289(2106)</td>
<td>F</td>
<td>151(1031)</td>
<td>211(1543)</td>
</tr>
<tr>
<td>Total Ranks</td>
<td>77.0</td>
<td>38.0</td>
<td>40.0</td>
<td>37.0</td>
<td>31.5</td>
<td>41.0</td>
<td>39.5</td>
<td>56.0</td>
</tr>
<tr>
<td>Total Normalized Ranks</td>
<td>2.44</td>
<td>1.21</td>
<td>1.27</td>
<td>1.17</td>
<td>1.00</td>
<td>1.30</td>
<td>1.25</td>
<td>1.78</td>
</tr>
</tbody>
</table>
### TABLE III
Computational Results Under Line Search LS2 ($\varepsilon = 0.1$)

<table>
<thead>
<tr>
<th>Problem (n)</th>
<th>(FR) ITER(IFN)</th>
<th>(HS) ITER(IFN)</th>
<th>(PR) ITER(IFN)</th>
<th>(P) ITER(IFN)</th>
<th>(P-SU) ITER(IFN)</th>
<th>MBFGS ITER(IFN)</th>
<th>MBFGS-SU ITER(IFN)</th>
<th>SDQN ITER(IFN)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 (n=2)</td>
<td>7(44)</td>
<td>5(38)</td>
<td>5(29)</td>
<td>5(34)</td>
<td>5(28)</td>
<td>5(34)</td>
<td>6(41)</td>
<td>4(28)</td>
</tr>
<tr>
<td>2 (n=2)</td>
<td>23(159)</td>
<td>27(189)</td>
<td>23(177)</td>
<td>20(151)</td>
<td>22(148)</td>
<td>20(146)</td>
<td>25(213)</td>
<td></td>
</tr>
<tr>
<td>3 (n=4)</td>
<td>F</td>
<td>72(457)</td>
<td>79(526)</td>
<td>72(465)</td>
<td>63(413)</td>
<td>95(629)</td>
<td>50(341)</td>
<td>63(450)</td>
</tr>
<tr>
<td>4 (n=4)</td>
<td>98(580)</td>
<td>27(198)</td>
<td>23(173)</td>
<td>20(142)</td>
<td>23(157)</td>
<td>29(205)</td>
<td>44(375)</td>
<td></td>
</tr>
<tr>
<td>5 (n=2)</td>
<td>11(88)</td>
<td>8(43)</td>
<td>9(63)</td>
<td>8(48)</td>
<td>9(57)</td>
<td>8(46)</td>
<td>8(56)</td>
<td></td>
</tr>
<tr>
<td>6 (n=5)</td>
<td>F</td>
<td>48(273)</td>
<td>42(209)</td>
<td>42(210)</td>
<td>48(253)</td>
<td>46(237)</td>
<td>44(266)</td>
<td></td>
</tr>
<tr>
<td>7 (n=20)</td>
<td>21(113)</td>
<td>16(97)</td>
<td>18(107)</td>
<td>15(92)</td>
<td>16(94)</td>
<td>16(91)</td>
<td>15(96)</td>
<td></td>
</tr>
<tr>
<td>8 (n=40)</td>
<td>31(157)</td>
<td>22(125)</td>
<td>23(128)</td>
<td>21(120)</td>
<td>23(129)</td>
<td>21(120)</td>
<td>21(125)</td>
<td></td>
</tr>
<tr>
<td>9 (n=100)</td>
<td>30(239)</td>
<td>34(179)</td>
<td>35(181)</td>
<td>33(170)</td>
<td>36(190)</td>
<td>34(175)</td>
<td>34(194)</td>
<td></td>
</tr>
<tr>
<td>10 (n=200)</td>
<td>72(327)</td>
<td>48(227)</td>
<td>48(230)</td>
<td>49(231)</td>
<td>47(225)</td>
<td>51(248)</td>
<td>48(227)</td>
<td>49(256)</td>
</tr>
<tr>
<td>11 (n=10)</td>
<td>248(1023)</td>
<td>24(128)</td>
<td>25(147)</td>
<td>21(120)</td>
<td>23(129)</td>
<td>21(120)</td>
<td>21(125)</td>
<td></td>
</tr>
<tr>
<td>12 (n=5)</td>
<td>208(989)</td>
<td>32(219)</td>
<td>51(299)</td>
<td>40(240)</td>
<td>46(289)</td>
<td>42(262)</td>
<td>40(289)</td>
<td></td>
</tr>
<tr>
<td>13 (n=10)</td>
<td>F</td>
<td>214(1554)</td>
<td>274(1872)</td>
<td>190(1377)</td>
<td>215(1613)</td>
<td>309(2244)</td>
<td>281(1925)</td>
<td>188(1462)</td>
</tr>
<tr>
<td>14 (n=10)</td>
<td>9(48)</td>
<td>7(40)</td>
<td>7(40)</td>
<td>7(40)</td>
<td>8(44)</td>
<td>9(49)</td>
<td>7(55)</td>
<td></td>
</tr>
<tr>
<td>15 (n=20)</td>
<td>15(74)</td>
<td>7(42)</td>
<td>7(42)</td>
<td>7(42)</td>
<td>9(50)</td>
<td>8(46)</td>
<td>10(55)</td>
<td>9(67)</td>
</tr>
<tr>
<td>16 (n=30)</td>
<td>25(116)</td>
<td>8(48)</td>
<td>8(48)</td>
<td>10(56)</td>
<td>10(56)</td>
<td>10(56)</td>
<td>10(74)</td>
<td></td>
</tr>
<tr>
<td>17 (n=50)</td>
<td>49(213)</td>
<td>11(61)</td>
<td>11(61)</td>
<td>12(65)</td>
<td>12(65)</td>
<td>15(77)</td>
<td>13(89)</td>
<td></td>
</tr>
</tbody>
</table>

Total Ranks:

<table>
<thead>
<tr>
<th>FR</th>
<th>HS</th>
<th>PR</th>
<th>P</th>
<th>P-SU</th>
<th>MBFGS</th>
<th>MBFGS-SU</th>
<th>SDQN</th>
</tr>
</thead>
<tbody>
<tr>
<td>79.0</td>
<td>35.0</td>
<td>51.5</td>
<td>41.5</td>
<td>28.5</td>
<td>39.0</td>
<td>35.0</td>
<td>50.5</td>
</tr>
</tbody>
</table>

Total Normalized Ranks:

<table>
<thead>
<tr>
<th>FR</th>
<th>HS</th>
<th>PR</th>
<th>P</th>
<th>P-SU</th>
<th>MBFGS</th>
<th>MBFGS-SU</th>
<th>SDQN</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.77</td>
<td>1.23</td>
<td>1.81</td>
<td>1.46</td>
<td>1.00</td>
<td>1.37</td>
<td>1.23</td>
<td>1.77</td>
</tr>
</tbody>
</table>
APPENDIX: TEST PROBLEMS AND STARTING SOLUTIONS

The following functions were used in our computational study, where $x^0$ is the initial point.

   \[ f(x) = (x_2 - x_1^2)^2 + 100(1 - x_1)^2, \quad x^0 = (-1 \cdot 2, 1 \cdot 0). \]

   \[ f(x) = 100(x_2 - x_1^3)^2 + (1 - x_1)^2, \quad x^0 = (-1 \cdot 2, 1 \cdot 0). \]

3. C. F. Wood’s Function (see Perry [15]).
   \[
   f(x) = 100(x_2 - x_1^2)^2 + (1 - x_1)^2 + 90(x_4 - x_3^2)^2 \\
   + (1 - x_3)^2 + 10 \cdot 1(x_2 - 1)^2 + 10 \cdot 1(x_4 - 1)^2 \\
   + 19 \cdot 8(x_2 - 1)(x_4 - 1), \quad x^0 = (-3, -1, -3, -1).
   \]

   \[
   f(x) = (x_1 + 10x_2)^2 + 5(x_3 - x_4)^2 + (x_2 - 2x_3)^4 \\
   + 10(x_1 - x_4)^4, \quad x^0 = (-3, -1, 0, 1).
   \]

   \[ f(x) = (x_2 - x_1^2)^2 + (x_1 - 1)^2, \quad x^0 = (-3, -1). \]

   \[
   f(x) = \sum_{i=1}^{n} 100(x_i - x_i^2)^2 + (1 - x_i)^2, \\
   x^0 = (-1 \cdot 2, 1 \cdot 0, -1 \cdot 2, 1 \cdot 0, ...).
   \]

7. Oren’s Power Function (Spedicato [22]).
   \[ f(x) = (x'Ax)^2, \quad A = \text{diag}(1, 2, ..., n), \quad x^0 = (1, 1, ..., 1). \]

   \[ f(x) = 100\{[x_3 - 10\theta(x_1, x_2)]^2 + [r(x_1, x_2) - 1]^2\} + x_3^2, \]
where

\[ 2\pi \theta(x_1, x_2) = \begin{cases} 
\arctan\left(\frac{x_2}{x_1}\right) & \text{if } x_1 > 0 \\
\pi + \arctan\left(\frac{x_2}{x_1}\right) & \text{if } x_1 < 0
\end{cases} \]

and

\[ r(x_1, x_2) = (x_1^2 + x_2^2)^{1/2}, \quad x^0(-1, 0, 0). \]

9. **Watson’s Function** (Kowalik and Osborne [8]).

\[ f(x) = \sum_{i=1}^{30} \left\{ \sum_{j=1}^{n} (j-1)x_i y_j^{1/2} - \left( \sum_{j=1}^{n} x_j y_j^{1/2} - 1 \right)^2 \right\} + x_1^2, \]

where

\[ y_i = \frac{i - 1}{29}, \quad x^0 = (0, 0, ..., 0). \]

10. **Mancino’s Function** (Spedicato [22]).

\[ f(x) = \sum_{i=1}^{n} f_i^2, \]

where

\[ f_i = \sum_{j=1}^{n} \left[ \left( x_i^2 + \frac{i}{j} \right)^{1/2} \left( \sin^2 \log \left( x_i^2 + \frac{i}{j} \right) \right)^{1/2} + \cos^2 \log \left( x_i^2 + \frac{i}{j} \right)^{1/2} \right] + \beta n x_i + \left( i - \frac{n}{2} \right)^{1/2} \]

and

\[ \alpha = 5, \beta = 14, \gamma = 3, \quad x^0 = (af_1(0), ..., af_n(0)), \]

with

\[ a = \frac{-\beta n}{\beta^2 n^2 - (\alpha + 1)^2(n - 1)^2}. \]

**ACKNOWLEDGMENTS**

The authors are grateful to an anonymous referee for some detailed, painstaking comments which significantly improved the presentation of this paper.
CONJUGATE GRADIENT METHODS

REFERENCES