New Generalized Conjugate Gradient Methods for the Non-Quadratic Model in Unconstrained Optimization

Abbas Al-Bayati

Received on 12/3/1996
Accepted for publication on 11/11/1996

Abstract

This paper presents two new conjugate gradient algorithms which use the non-quadratic model in unconstrained optimization. The first is a new generalized self-scaling variable metric algorithm based on the Sloboda generalized conjugate gradient method which is invariant to a nonlinear scaling of a strictly convex quadratic function; the second is an interleaving between the generalized Sloboda method and the first algorithm: all these algorithms use exact line searches.

Numerical comparisons over twenty test functions show that the interleaving algorithm is best overall and requires only about half the function evaluations of the Sloboda method.

Interleaving algorithms are likely to be preferred when the dimensionality of the problem is increased.

Key words: Unconstrained Optimization, Conjugate Gradient Methods, Variable Metric Methods, Exact Line Search, Non-quadratic Model.

1. Introduction

Let \( f = f(x) \) be a continuously differentiable function of \( x \in \mathbb{R}^n \) with gradient \( g = g(x) \). The problem of finding a minimizer \( x_{\text{min}} \) of \( f \) is often tackled by conjugate gradient (CG) algorithms which are based on the following iterative scheme:

\[
x^{+} = x^+ + \lambda d
\]  

(1)

© 2001 by Yarmouk University, Irbid, Jordan.
* Professor of Numerical Optimization. Department of Mathematics, College of Computers and Mathematical Science, University of Mosul, Iraq
Where $\mathbf{x}$ is the initial estimate of $x_{\text{mm}}$, $x^*$ is the new estimate; $\lambda$ is a positive scalar chosen by exact line search (ELS) so that:

$$f(x^*) = \min \lambda \cdot f(x - \lambda \cdot d)$$

(2)

and $d$ is the search direction. In fact the linear search on $d$ is performed iteratively over the direction set $d_1, d_2, \ldots$. So that at the $k$-th iteration

$$d_1 = -g_1 \quad \text{(the gradient of at } x_1); \quad k = 0$$

$$d_k = -g_k + \beta d \quad \text{for } k \geq 1$$

(3)

where $\beta$ is a scalar chosen to ensure $d_k$ is conjugate to $d$ with respect to the Hessian matrix of $f(x)$. Different choices are possible for $\beta$; a commonly used one due to Hestenes and Stiefel (1952) is:

$$\beta = (y^T \cdot g_k) / (y^T \cdot d_k)$$

(4)

where

$$y = g_k \cdot g$$

(5)

Section 2 of this paper contains some definitions and the essential abbreviations of the basic results Tassopoulous; Boland; Al-Bayati and Sloboda in the field of non-quadratic models. Section 3 contains some basic known results of Sloboda with Sloboda’s original algorithm (Algorithm I). Section 4 presents Al-Bayati’s modification of the Oren’s formula in (33) with the first new algorithm (Algorithm II) followed by the second new algorithm (Algorithm III) which modifies the known results of Buckley. Finally, section 5 presents all the numerical results of this work.

**Basic results and definitions:**

The rate of convergence of a variety of CG-algorithm has been investigated by many authors; the most general results are given by Baptist and Store (1977) and Store (1977) where it is also shown that the algorithms with ELS have the property of $n$-step quadratic convergence. To improve both the efficiency and the local rate of convergence of CG-algorithm several techniques have been suggested by Dixon (1975) and Sloboda (1979). In order to improve the global rate of convergence of CG-algorithms it is necessary to construct special algorithms for more general
functions than the quadratic. In series of papers: Fried (1971); Boland et al (1979a), (1979b), (1979c); Tassopoulous (1984); Al-Bayati (1993); Al-Bayati et al. (1992), (1994), (1995) various algorithms have been suggested which are efficient for special non-quadratic models.

A generalized CG-algorithm is suggested by Sloboda (1980) for the model
\[ F(x) = Z(F(x)) \]  \hspace{1cm} (6)
Where \( Z: \mathbb{R}^1 \to \mathbb{R}^1 \) is a differentiable function with non-vanishing derivatives and \( F: \mathbb{R}^n \to \mathbb{R}^1 \) is a strictly convex quadratic function provided that the functions \( F \) and \( Z \) are known a priori.

A function of the form (6) is called a nonlinear scaling of \( F(x) \), and an algorithm which generates the same sequence of points and direction vectors when applied to function \( f \) as to function \( F \) is defined as "invariant to nonlinear scaling \( F(x) \)" by Specicato (1976).

We also require that the nonlinear scaling satisfies the following condition: a function \( Z \) is nonlinear scaling of \( F(x) \) and
\[ \frac{dZ}{dF} > 0; \text{ for } x \neq x_{\text{min}} \] \hspace{1cm} (7)

**Property 1:** It follows from the definitions that, for a \( Z \) which is a nonlinear scaling of \( F \) (see Boland et al (1979a)).

(One) Every contour line of \( Z \) is a contour line of \( F \), and vice versa.

(Two) \( Z \) has no more stationary points than \( F \).

(Three) If \( x_{\text{min}} \) is a minimizer of \( F \), then it is also a minimizer of \( Z \).

(Four) The sequence of points which minimize \( F \) does the same for \( Z \).

**Known theoretical results:**

Sloboda (1980) proposed a generalized CG-method which is invariant to a particular nonlinear scaling of \( F(x) \). Let \( f \) be a strictly convex function of the form (6). Denoting \( g \) as the gradient of \( f \) and \( G \) the gradient of \( F \) and then differentiating (6) with respect to \( x \) we get:
\[ g = cG, \ c \neq 0 \] \hspace{1cm} (8)
Where

\[ c = (dZ/dF) \]  

(9)

For the ELS clearly

\[ d^T G* = d^T G = 0 \]  

(10)

Sloboda defined the following linear function

\[ L_k (\lambda) = (\lambda / \lambda_k) (d^T G* - d^T G) - d^T G \]  

(11)

Where

\[ \lambda \in (0, \lambda_k); \lambda_k \neq 0 \]  

(12)

and G is considered to be known, hence he has proved the following lemma:

**Lemma 1:**

Define \( G^- = G(x - \lambda d/2) \);

(13)

Then for the standard CG-method defined in (3)-(5) it can be shown that

\[ G* = 2G^- - G \]  

(14)

The proof can be found in Sloboda (1980)

Now to express \( G^- \) in terms of \( g^- \) put \( \lambda = \lambda_k/2 \) in (11) to get:

\[
L_k (0.5 \lambda_k) = (0.5 \lambda / \lambda_k) (d^T G* - d^T G) - d^T G \\
= (0.5) d^T G - 0.5 d^T G + d^T G \\
= 0.5 d^T G + 0.5 d^T G
\]  

(15)

and using (14) yields

\[ L_k (0.5 \lambda_k) = d^T G^- \]  

(16)

Substituting for \( G^- \) from (14) and assuming ELS we get:

\[ L_k (0.5 \lambda_k) = 0.5 d^T G \]  

(17)

And so from (16) and (17) we have

\[ 0.5 d^T G = d^T G^- \]  

(18)

Again, from (8) we have
\[ d^T \cdot g^r = c^+ d^T \cdot G^- \]  \hfill (19)

(which define \( c^+ \)), so that

\[ c^+ = (d^T \cdot g^r)/(d^T \cdot G^-) \]  \hfill (20)

But

\[
G^- = \frac{1}{c^+} g^r = [(d^T \cdot G^-)/(d^T \cdot g^r)]g^r
\]

\[
= [(0.5 d^T \cdot G)/(d^T \cdot g^r)]g^r.
\]  \hfill (21)

Thus by use of this linearization technique and with (14), Sloboda's generalized CG-method can be expressed as follows:

Algorithm 1:

For a strictly convex function \( f \in C^2 \) with gradient \( g \):

choose \( x_1 \in \mathbb{R}^n \) and proceed as follows: (see Sloboda (1980)).

Step 1: set \( k = 1 \); \( g_1 = g_1 \) and \( d_1 = -g_1 \).

Step 2: compute \( \lambda \) by ELS and set \( x_1 = x_1 + \lambda \cdot d \).

Step 3: compute \( g^{+} = g \cdot (x_1 - \lambda \cdot d/2) \).

Step 4: test for convergence: if achieved stop, if not continue.

Step 5: if \( k = 0 \mod (n) \) go to step 1 else continue.

Step 6: compute \( \tilde{g}_s = \tilde{c} \cdot g^{+} - g \) where \( \tilde{c} = (d^T \cdot \tilde{g} / d^T \cdot g^{+}) \).

Step 7: compute the new search direction

\[ d_1 = -\tilde{g}_s + \beta d \] where \( \beta = (\tilde{y} \cdot \tilde{g}_s / d^T \cdot \tilde{y}); \tilde{y} = \tilde{g}_s - \tilde{g} \).

Step 8: set \( k = k + 1 \) and go to step 2.

Algorithm 1 terminates after \( n \) iterations in the case of a nonlinear scaled quadratic function using ELS.
**Theorem 1:**

Algorithm I is invariant to a nonlinear scaling of the function $F(x)$. For proof see Sloboda (1980).

Variable metric (VM) or quasi-Newton (QN) are iterative methods of the form:

$$x_\ast = x - \lambda_\ast H_g$$  \hspace{1cm} (22)

Where $\lambda_\ast$ is a scalar chosen either by ILS so that

$$f(x_\ast) < f(x)$$  \hspace{1cm} (23)

Or by ELS (so that (23) is also satisfied). $H$ is an $n \times n$ matrix intended as an approximation to the inverse Hessian. $H_1$ is given initially, and in the well-known Broyden (1970) one-parameter update $H$ is modified at every iteration according to the scheme

$$H_\ast = H + \frac{b + \theta a}{b^2} vv^T + \frac{\theta - 1}{a} Hyy^T H - \frac{\theta}{b} (Hyy^T + vv^T H)$$  \hspace{1cm} (24)

where $\theta$ is a free parameter and we define

$$v = x_\ast - x$$  \hspace{1cm} (25)

$$a = y^T H y$$  \hspace{1cm} (26)

$$b = v^T y$$  \hspace{1cm} (27)

It is possible to rewrite (24) as

$$H_\ast = U(H, v, y).$$  \hspace{1cm} (28)

In studying the effects of scaling the quadratic function $F$ for VM-methods we recall that (22) is not invariant to a linear scaling of $F$ when (24) is used, in that, unless $H_1$ is also suitably scaled, the sequence of generated points changes if $F$ is multiplied by a scalar. This remark, made originally by Bard (1968), has motivated Oren (1972) in developing a class of updates invariant to linear scaling.

**Main Theoretical Results:**

Oren's class is obtained by introducing in (24) a new parameter $\mu$ which multiplies the matrix $H$. Oren chooses the parameter in the following way:

$$\mu = (\varepsilon / b) \beta + (b/a) (1 - \beta); \quad 0 \leq \beta \leq 1$$  \hspace{1cm} (29)
where
\[ \varepsilon = \nabla^T H \nabla = \lambda^2 g^T H g, \]  
(30)

and \( a, b \) are defined earlier in (26) and (27); thus Oren’s update is given by
\[ H_{\text{oren}} = \mu H + \frac{b + \theta a}{b^T} \nabla \nabla^T + \frac{\mu(\theta - 1)}{a} H y y^T H - \frac{\mu \theta}{b} (H y v^T + v y^T H) \]  
(31)

The introduction of the new parameter also improves the overall stability and convergence properties of the update as described by Oren (1972) and by Oren & Spedicato (1976). A more general scaling has been considered by Spedicato (1976); such a scaling transforms \( F \) into a new function where (7) is satisfied. He shows that the sequence of points generated is invariant with respect to nonlinear scaling if:
\[ y = \frac{g^*}{dZ / dF} - \frac{g}{dZ / dF} \]  
(32)

However, this type of scaling also uses functions for which the analytic form is known a priori.

Al-Bayati (1991) introduced another new family of self-scaling VM-methods given by:
\[ H_* = H + (\sigma + a / b) \frac{y v^T}{b} + \frac{\theta - 1}{a} H y y^T H - \frac{\theta}{b} (H y v^T + v y^T H) \]  
(33)

where \( \theta \) is again a free parameter; and
\[ \sigma = 1 / \mu \]  
(34)

For \( \theta = 1 \), (33) reduces to
\[ H_* = H + \left( \sigma + a / b \right) \frac{y v^T}{b} - \frac{1}{b} (H y v^T + v y^T H) \]  
(35)

If an estimate of the inverse Hessian is maintained (rather than an estimate of the Hessian itself which is sometimes preferred) then there is a strong motivation for choosing \( \beta = 0 \) in (29), namely, that \( H^{-1} \) is not required; this gives:
\[ \sigma = a / b \]  
(36)
However, it is possible to generalize Al-Bayati's new family of self-scaling VM-updates (35) to be invariant to a nonlinear scaling (assuming that the functions are known a priori) by the following algorithm:

**Algorithm II:**

Let $f$ be any non linear scaling of the function $F(x)$; $x_0 \in \mathbb{R}^n$ and $H_i$ is any positive definite nxn matrix (though usually $H_i=I$).

Step 1: set $k=1$; $\bar{g}_i = g_1$ and $d_1 = -H_1 g_1$.

Step 2: compute $x^* = x + \lambda d$; $\lambda$ determined by ELS.

Step 3: set $\bar{g}_i = \omega g^* - \bar{g}_i$; $\omega = \left( d^T g^* / d^T g^* \right)$ and $g^* = g(x - \lambda d / 2)$.

Step 4: test for convergence; if not continue.

Step 5: update the matrix $H$ using (35), but with $y = \bar{y}$.

Step 6: compute $d_1 = -H_1 \bar{g}_i$.

Step 7: set $k = k + 1$ and go to step 2.

To prove that the above new algorithm is invariant to a nonlinear scaling of the function $F(x)$, consider the following theorem:

**Theorem II:** Algorithm II is invariant to a nonlinear scaling of $F(x)$ with ELS.

**Proof:** Let $\{x\}$, $\{G\}$, $\{H\}$ be the sequences obtained by minimizing the quadratic function $F(x)$ and let $\{x\}$, $\{g\}$, $\{H\}$ be the sequences obtained by minimizing the general function $f(x)$. To satisfy the definition in section 2 we shall assume that the initial values are identical:

$$x_1 = \bar{x}_1; \quad H_1 = H_1; \quad \bar{g}_1 = g_1 = G_1; \quad (37)$$

without loss of generality

$$x_2 = x_1 + \lambda_1 d_1 \quad (38)$$
and using (18), (19)

\[ \omega_i = \frac{d_i^T g_1^*}{d_i^T g_1^*} = \frac{d_i^T G_i}{0.5c_i^* d_i^T G_i} = \frac{2}{c_i^*} \]  \hspace{1cm} (39)

Also

\[ \tilde{g}_2 = \left(2 / c_i^*\right) g_1^* - \tilde{g}_1 \]

\[ = 2G_i^* - \tilde{g}_i \]

from (8)

\[ = 2G_i^* - \tilde{G}_i \]

from (37)

\[ = G_2 \]

from (14)

therefore

\[ \tilde{g}_2 = G_2 \]  \hspace{1cm} (40)

To complete by induction, we suppose that:

\[ x_j = x_j^* ; \tilde{g}_j = G_j \text{ and } H_j = H_j^* \text{ for } j=1,2, \ldots, k. \]  \hspace{1cm} (41)

Then for \( k+1 \)

\[ \tilde{g}_{k+1} = \left(2 / c_k^*\right) g_k^* - g_k^* \]

from (8)

\[ = 2G_k^* - G_k \]

from (19) and (41)

\[ = G_{k+1} \]

from (41)
Therefore,

\[ x_{k+1} = x_{k+1}' \quad \text{by using ELS}, \]

and hence

\[ H_{k+1} = U \left( H_k, x_{k+1} - x_k, g_{k+1} - g_k \right) \]

\[ = U \left( H'_k, x'_{k+1} - x'_k, G_{k+1} - G_k \right) \]

\[ = H'_{k+1}. \]

Thus the search directions at \( x_k \) for \( k=1,2, \ldots \) are the same as at \( x'_k \) for \( k = 1, 2, \ldots \), and hence the proof is established.

Both CG and VM-methods are relatively successful at minimizing "smooth" nonlinear functions of several variables. In particular, CG-methods require much less storage to implement than VM or QN algorithms and are therefore preferred when storage limitations occur. However, they have a slower rate of convergence so there have been some attempts to combine CG and QN algorithms in order to obtain good convergence properties and low storage requirements. The most successful of these algorithms are: Shanno and Phua (1978) and Buckley (1978a), (1978b).

In this section we shall describe also another new algorithm which effectively interleaves CG and VM-steps. It is also related to one given originally by Buckley (1978a), but our implementation differs in that we use the scaled quadratic model instead of the quadratic itself. Moreover, the new algorithm uses highly accurate line searches. Numerical results are presented in the next section to demonstrate that the use of VM-steps can indeed increase the rate of convergence of generalized Sloboda’s CG-method.

The new algorithm is a Combination algorithm between CG-algorithm I and the generalized VM-algorithm II. The primary objective here is to show that, using Al-Bayati's self-scaling VM-update (35), the sequence of points generated is the same as in the generalized CG-algorithm I. Hence the property of finite termination (in at most \( n \) steps) is preserved.
Before making a few more observations we shall outline briefly the proposed strategy for the new interleaved generalized CG-VM method.

**Algorithm III**

Let $f$ be a nonlinear scaling of the quadratic function $F$; given $x_i$ and a matrix $H_i = I$ set $\tilde{g}_i = g_i$, $i=1$ and $t=1$ initially.

Step 1: set $d_i = -H_i \tilde{g}_i$.

Step 2: for $k=t, t+1, t+2, ...$ iterate with

$$x_{k+1} = x_k + \lambda_k d_k$$

$$\beta_k = \left( g_k \mathbf{T} H_k y_k \right) / \left( d_k \mathbf{T} \tilde{y}_k \right)$$

$$d_{k+1} = -H_k \tilde{g}_{k+1} + \beta_k d_k$$

$$\omega_k = d_k \mathbf{T} \tilde{g}_k / d_k \mathbf{T} \tilde{g}_k^+$$

$$g_k^+ = g(x_k - \lambda_k d_k / 2)$$

$$\tilde{g}_{k+1} = \omega_k g_k^+ - \tilde{g}_k$$

$$\tilde{y} = \tilde{g} - \tilde{g}$$

When the algorithm is not converged, until a restart.

Step 3: if a restart is indicated namely with the Powell (1977) restarting criterion is satisfied, i.e.

$$\left| g \mathbf{T} g \right| \geq 0.2 \left| g \mathbf{T} g * \right|$$

then reset $t$ to the current $k$. Update $H_i$ by

$$H_{i+1} = H_i - \left( H_i y_i v_i \right)/b_i + v_i \left[ 2a_i / b_i \right] v_i - \left( H_i y_i / b_i \right)$$
Al-Bayati

Step 4: replace \( i \) by \( i+1 \) and repeat from (43).

In order to implement algorithm III a number of important features must be considered:

(a) Line search: The line search routine employed is a cub interpolation which uses function values and their gradients. The actual program used in a adaptation of the routine published by Bunday (1984). This routine proceeds as follows: from a starting point \( x \) a search is conducted along direction \( d \) for a "better" point \( x^* = x + \lambda d \), for some suitable value of \( \lambda \). If an ELS is performed then \( \lambda \) is required to minimize \( f \) such that

\[
d^\top g_* = 0 \tag{47}
\]

In practice, however, the directional derivative in (47) is reduced significantly but no necessarily to zero. Suitable conditions on \( \lambda \) have been proposed by many author in order to prevent oscillatory behavior. In particular, Powell (1976) imposed the following conditions:

\[
f_* - f \leq \eta_1 \lambda |d^\top g|
\]

and, to ensure that \( \lambda \) is not too small, he assumed that

\[
|d^\top g_*| < \eta_2 |d^\top g|
\]

is also satisfied, where \( \eta_1 \) and \( \eta_2 \) are constants that satisfy the bounds

\[
0 < \eta_1 < \eta_2 \quad \text{and} \quad \eta |< 0.5
\]

for example \( \eta_1 = 0.0001 \) and \( \eta_2 = 0.01 \) are quoted as suitable values to define an accurate line search. Powell also pointed out that inequalities (49) and (50) ensure that \( v^\top y > 0 \).

(ii) Storage requirement of the update: First we observed that Al-Bayati's VM-update (35) can be expressed as:

\[
H_{i+1} = H_i - \left( S_i v_i^\top \right) / b_i + v_i \left( \frac{2a_i}{b_i^2} \right) v_i^\top - S_i / b_i^\top \tag{51}
\]

where

\[
S_i = H_i y_i
\]

-20-
and a, b, are scalars defined earlier in (26) and (27). It is clear that for each update we require (2n−2)storage locations in order to store a, b, s, and v. For m update we therefore need m(2n−2) storage locations which may be consider as moderate storage, intermediate between the VM and CG requirements.

**Numerical results:**

Comparative tests were preformed with twenty well-known test functions. All the results were obtained with newly-programmed FORTRAN routines which employ double precision. The comparative performances of the algorithms are in the usual way by considering both the total number of functions evaluations(NOIF) and total number of iterations (NOII): in each case the convergence criterion is that the value of f(x) is reduced below 5x10^{-10}. The cubic fitting technique, published in its original form by Bunday (1984) is used as the common linear search subprogram.

Each of the function was solved using the following three algorithms:

1. Algorithm I: (the original) algorithm published by Sloboda) call it SLOBODA.
2. Algorithm II (the new variable metric version of Sloboda for non-quadratic) call it VMSLOB.
3. Algorithm III (the new combined form of algorithm I and II) call it CVMSLOB.

All the results for the 60 problem-algorithm combinations are presented in Table1.

Sloboda himself did not perform numerical trials on his algorithm: nevertheless, his claims that his algorithm, being a generalized CG-method is superior to standard CG-methods is fully justified by these results. We are therefore using the Sloboda method as the standard against which to measure the two new algorithms: the first and third algorithms include the (generally beneficial ) restart after every n iterations whereas the second (VMSLOB) does not.

The values of NOIF and NOII are given in the table for each algorithm, summing over the full set of mixed-dimensionality test function. Using Sloboda as the 100% standard it is clear that VMSLOB saves about 37% in NOIF and 40% in NOII and that CVMSLOB saves about 47% in either NOIF or NOII on the same measures! However, detailed examination shows that the Sloboda algorithm I beats one of the other algorithms (in either NOIF or NOII) in seven cases out of the possible fourty.
The table also shows that the number of VM-updates (NOU) required by algorithm III is only 229 so that it uses only 195 CG-steps to minimize the total of twenty test functions.

Finally, paired t-test is applied to test for significance level between algorithm I and the two new algorithms II, III. This test indicated that there is a highly significant difference between the proposed new algorithms and Sloboda's original algorithm. The significance level and the t-values are shown below:

For algorithm I and III:
NOF: t=-3.24  S.level= 0.009
NOI: t=-2.16  S.level= 0.022

For algorithm II and III:
NOF: t=-3.11  S.level= 0.002
NOI: t=-3.01  S.level= 0.003

The conclusion, on the basis of numerical tests, is that interleaved algorithms may offer significant advantages for high dimensionality problems.

<table>
<thead>
<tr>
<th>Test Function</th>
<th>N</th>
<th>Sloboda (I) NOI(NOF)</th>
<th>VM-SLOB (I) NOI(NOF)</th>
<th>CV-MLOB (III) NOI(NOF)</th>
<th>NOI</th>
</tr>
</thead>
<tbody>
<tr>
<td>ROSSLN</td>
<td>2</td>
<td>33 (85)</td>
<td>17 (49)</td>
<td>21 (57)</td>
<td>17</td>
</tr>
<tr>
<td>CUBIC</td>
<td>2</td>
<td>18 (52)</td>
<td>9 (28)</td>
<td>9 (30)</td>
<td>9</td>
</tr>
<tr>
<td>BEALE</td>
<td>2</td>
<td>10 (26)</td>
<td>7 (17)</td>
<td>8 (20)</td>
<td>6</td>
</tr>
<tr>
<td>FRIAD</td>
<td>2</td>
<td>8 (22)</td>
<td>6 (17)</td>
<td>6 (17)</td>
<td>6</td>
</tr>
<tr>
<td>BIGGS</td>
<td>3</td>
<td>21 (78)</td>
<td>11 (36)</td>
<td>12 (37)</td>
<td>7</td>
</tr>
<tr>
<td>POWEL3</td>
<td>3</td>
<td>15 (24)</td>
<td>12 (26)</td>
<td>10 (23)</td>
<td>7</td>
</tr>
<tr>
<td>POWEL4</td>
<td>4</td>
<td>51 (130)</td>
<td>40 (90)</td>
<td>26 (59)</td>
<td>19</td>
</tr>
<tr>
<td>WOOD</td>
<td>4</td>
<td>33 (80)</td>
<td>28 (83)</td>
<td>25 (66)</td>
<td>16</td>
</tr>
<tr>
<td>DIXON</td>
<td>10</td>
<td>19 (41)</td>
<td>20 (43)</td>
<td>17 (37)</td>
<td>1</td>
</tr>
<tr>
<td>OREN</td>
<td>10</td>
<td>19 (92)</td>
<td>16 (81)</td>
<td>11 (70)</td>
<td>8</td>
</tr>
<tr>
<td>EX-POWEL1</td>
<td>20</td>
<td>60 (157)</td>
<td>42 (101)</td>
<td>34 (75)</td>
<td>24</td>
</tr>
<tr>
<td>EX-WOOD</td>
<td>20</td>
<td>48 (110)</td>
<td>29 (85)</td>
<td>25 (66)</td>
<td>16</td>
</tr>
<tr>
<td>SHALLOW</td>
<td>40</td>
<td>8 (20)</td>
<td>8 (19)</td>
<td>8 (19)</td>
<td>5</td>
</tr>
<tr>
<td>FULL</td>
<td>40</td>
<td>42 (85)</td>
<td>38 (77)</td>
<td>42 (85)</td>
<td>0</td>
</tr>
<tr>
<td>EX-ROSENF</td>
<td>50</td>
<td>17 (49)</td>
<td>17 (49)</td>
<td>21 (57)</td>
<td>17</td>
</tr>
<tr>
<td>EX-WOOD</td>
<td>60</td>
<td>109 (231)</td>
<td>39 (87)</td>
<td>25 (66)</td>
<td>16</td>
</tr>
<tr>
<td>WOLF</td>
<td>80</td>
<td>52 (105)</td>
<td>47 (95)</td>
<td>40 (81)</td>
<td>3</td>
</tr>
<tr>
<td>NON-DIGN</td>
<td>90</td>
<td>19 (57)</td>
<td>19 (57)</td>
<td>22 (62)</td>
<td>17</td>
</tr>
<tr>
<td>EX-POWEL1.1</td>
<td>100</td>
<td>106 (303)</td>
<td>59 (148)</td>
<td>37 (81)</td>
<td>25</td>
</tr>
<tr>
<td>EX-WOOD</td>
<td>100</td>
<td>119 (251)</td>
<td>31 (89)</td>
<td>25 (66)</td>
<td>16</td>
</tr>
<tr>
<td>(NOI) TOTAL</td>
<td>807</td>
<td>480</td>
<td>424</td>
<td>229</td>
<td>229</td>
</tr>
<tr>
<td>(NOI)</td>
<td>2008</td>
<td>1277</td>
<td>1074</td>
<td>***</td>
<td></td>
</tr>
</tbody>
</table>

-22-
PERFORMANCE OF THE NEW ALGORITHMS IN RELATION TO SLOBODA'S
ALGORITHM (I)

<table>
<thead>
<tr>
<th></th>
<th>SLOBODA (I)</th>
<th>VMSLOB (II)</th>
<th>CVMSLOB (II)</th>
</tr>
</thead>
<tbody>
<tr>
<td>NO1</td>
<td>100</td>
<td>60.2</td>
<td>52.5</td>
</tr>
<tr>
<td>NO2</td>
<td>100</td>
<td>63.6</td>
<td>53.5</td>
</tr>
</tbody>
</table>

ALL THE ALGORITHMS TERMINATED WHEN $|f - f_{\min}| < 5 \times 10^{-10}$

References


