JOURNAL OF MATHEMATICAL ANALYSIS AND APPLICATIONS **197**, 586–607 (1996)
ARTICLE NO. 0041

A New Unconstrained Optimization Method for Imprecise Function and Gradient Values

M. N. Vrahatis, G. S. Androulakis, and G. E. Manoussakis

Department of Mathematics, University of Patras, GR-261.10 Patras, Greece

Received June 13, 1994

A new algorithm for unconstrained optimization is presented which is based on a modified one-dimensional bisection method. The algorithm actually uses only the signs of function and gradient values. Thus it can be applied to problems with imprecise function and gradient values. It converges in one iteration on quadratic functions of n variables, it rapidly minimizes general functions and it does not require evaluation or estimation of the matrix of second partial derivatives. The algorithm has been implemented and tested. Performance information for well-known test functions is reported. © 1996 Academic Press, Inc.

1. INTRODUCTION

There is no need to emphasize the importance of finding the extrema of a function

$$f\colon \mathscr{D} \subset \mathbb{R}^n \to \mathbb{R}, \tag{1.1}$$

where $\mathbb{R}^n$ indicates the real Euclidean n-space, and the numerous applications in many different fields such as mathematics, statistics, operation research, management science, economics, computer science, engineering, and physical sciences.

Optimization methods (non-numerical ones) existed even before Newton, Lagrange, and Cauchy. After the contributions of Bernoulli, Euler, Lagrange, and Weierstrass to the calculus of variations, and the use of certain multipliers by Lagrange to solve the constrained minimization problem, Cauchy made the first application of the *steepest descent method* to solve unconstrained minimization problems [4].

A well-known method in the class of steepest descent methods [11, 2, 18] for unconstrained minimization of functions (1.1) having Lipschitz continuous first partial derivatives is given by Armijo in [1], which under some suitable assumptions on f and $\mathscr{D}$ always converges to a local minimum.

This method is in fact a modification of Cauchy's method. It allows for the possibility of variable step size and does not require knowledge of the value of the Lipschitz constant.

Moreover, there is a class of methods called *nonlinear conjugate gradient methods*, as typified by the *Fletcher–Reeves* (FR) *algorithm* [8, 11, 2, 18, 17] and the closely related *Polak–Ribiere* (PR) *algorithm* [16, 17]. The convergence of these methods, for general nonlinear problems, is linear [13, 11]. Conjugate gradient methods require storage of order only a fewtimes n. On the other hand, these methods require derivative calculations as well as one-dimensional subminimization and they are very sensitive to rounding off errors.

Another efficient class of methods is known under the names *quasi-Newton* and *variable metric methods*, as typified by the *Davidon–Fletcher–Powell* (DFP) *algorithm* [5, 7, 11, 2, 18, 17] (sometimes referred to as *Fletcher–Powell* (FP) *algorithm*) or the closely related *Broyden–Fletcher–Goldfarb–Shanno* (BFGS) *algorithm*. The above-mentioned methods are very stable and they converge superlinearly [13]. On the other hand, these methods require storage of order n^2 and they require derivative calculations. Besides, they approximate the inverse of the Hessian matrix and they require one-dimensional subminimization.

All the above mentioned methods require precise function and gradient values. However, in many optimization problems of practical interest the values of the objective functions are known only imprecisely. For example, when the function and gradient values depend on the results of numerical simulations, then it may be difficult or impossible to obtain very precise values. Or, in other cases, it may be necessary to integrate numerically a system of differential equations in order to obtain a function value, so that the precision of the computed value is limited.

In this contribution we describe a new efficient numerical method for computing an unconstrained local minimum which needs neither information about the Hessian matrix nor matrix inversions. This method is based on a modified one-dimensional bisection method and it actually requires only the signs of function and gradient values to be correct. Consequently, it can be applied to problems with imprecise function values. The algorithm converges in one iteration on quadratic functions of n variables and, as numerical results indicate, rapidly minimizes general functions. Comparing the new method with the previously mentioned methods to a variety of test functions we obtain very promising results.

In the next section we give the theoretical background of our method. Also in this section we present a modified bisection method for the computation of a root of a single continuous function within a given interval $(a, b) \subset \mathbb{R}$. Then, in Section 3, we describe the new algorithm and we also study its convergence. Applications of this algorithm are presented in Section 4. We finally end, in Section 5, with some concluding remarks and a short discussion for further research.

2. THEORETICAL BACKGROUND

In this section we shall give the theoretical background on which our method is based.

(a) *Armijo's Modified Steepest Descent Algorithm*

First, let us give Armijo's modified steepest descent algorithm. To do this the following assumptions are needed:

(a) The function f is a real-valued function defined and continuous everywhere in $\mathbb{R}^n$, bounded below in $\mathbb{R}^n$,

(b) for fixed point $x^0 \in \mathbb{R}^n$, $\mathscr{S}(x^0) = \{x\colon f(x) \le f(x^0)\}$, then $f \in C^1$ on $\mathscr{S}(x^0)$ and ∇f is Lipschitz continuous on $\mathscr{S}(x^0)$ with constant $K > 0$,

(c) $r > 0$ implies that $m(r) > 0$, where $m(r) = \inf_{x \in \mathscr{S}_r(x^0)} \|\nabla f(x)\|$, $\mathscr{S}_r(x^0) = \mathscr{S}_r \cap \mathscr{S}(x^0)$, $\mathscr{S}_r = \{x\colon \|x - x^*\| \ge r\}$, and x^* is any point for which $f(x^*) = \inf_{x \in \mathbb{R}^n} f(x)$, (if $\mathscr{S}_r(x^0)$ is void, we define $m(r) = \infty$).

The well-known *Cauchy's method* or *the steepest descent algorithm* [4] states that, if the above assumptions are fulfilled, then the sequence $\{x^k\}_{k=0}^{\infty}$ defined by

$$x^{k+1} = x^k - \frac{1}{2K} \nabla f(x^k), \qquad k = 0, 1, 2, \dots, \tag{2.1}$$

converges to the point x^* which minimizes f.

In [1] Armijo modified Cauchy's method to the following one, called *Armijo's method* or *the modified steepest descent algorithm*. According to this method, if the above assumptions are fulfilled and $\eta_m = \eta/2^{m-1}$, $m = 1, 2, \dots,$ with η an arbitrary assigned positive number, then the sequence $\{x^k\}_{k=0}^{\infty}$ defined by

$$x^{k+1} = x^k - \eta_{m_k} \nabla f(x^k), \qquad k = 0, 1, 2, \dots, \tag{2.2}$$

where m_k is the smallest positive integer for which

$$f\bigl(x^k - \eta_{m_k} \nabla f(x^k)\bigr) - f(x^k) \le -\tfrac{1}{2} \eta_{m_k} \|\nabla f(x^k)\|^2, \tag{2.3}$$

converges to the point x^* which minimizes f.

Based on this, we are able to give the following subprocedure, which we are going to use in the sequel, where *MAR* is the maximum number of Armijo iterations required, η is an arbitrary assigned positive number, and ε the predetermined desired accuracy.

ALGORITHM 2.1. *Armijo's Modified Steepest Descent Algorithm.*

```
Step 1  Input{$f$; $x^0$; $MAR$; $\eta_0$; $\varepsilon$}.
Step 2  Set $k = -1$.
Step 3  If $k < MAR$, replace $k$ by $k + 1$, set $\eta = \eta_0, m = 1$ and go to
        next step; otherwise, go to Step 8.
Step 4  If $(x^k - \eta \nabla f(x^k)) - f(x^k) \leq -\frac{1}{2}\eta \|\nabla f(x^k)\|^2$, go to Step 6; oth-
        erwise, set $m = m + 1$ and go to next step.
Step 5  Set $\eta = \eta / 2^{m-1}$ and return to Step 4.
Step 6  Set $x^{k+1} = x^k - \eta \nabla f(x^k)$.
Step 7  If $\|\nabla f(x^k)\| \leq \varepsilon$, go to Step 8; otherwise go to Step 3.
Step 8  Output{$x^k$; $f(x^k)$; $\nabla f(x^k)$}.
```

(b) *Nonlinear* SOR

There is a class of methods for the numerical solution of a system of nonlinear equations

$$F(x) = \Theta^n = (0, 0, \ldots, 0), \tag{2.4}$$

where $F = (f_1, \ldots, f_n): \mathscr{D} \subset \mathbb{R}^n \to \mathbb{R}^n$ is a continuously differentiable mapping on an open neighborhood $\mathscr{D}^* \subset \mathscr{D}$ of a solution $x^* \in \mathscr{D}$, which arise from iterative procedures used for systems of linear equations [21, 13]. The main advantage of these methods is the reduction to simpler one-dimensional nonlinear equations for the components $f_1, f_2, \ldots, f_n$ of F. The best-known method of this type is the *nonlinear successive overrelaxation* (nonlinear SOR) method which solves at the $(p+1)$st iteration the one-dimensional equation

$$f_i\left(x_1^{p+1}, \ldots, x_{i-1}^{p+1}, x_i, x_{i+1}^p, \ldots, x_n^p\right) = 0, \tag{2.5}$$

for x_i and then sets

$$x_i^{p+1} = x_i^p + \omega_p\left(x_i - x_i^p\right), \qquad i = 1, \ldots, n, \qquad p = 0, 1, \ldots. \tag{2.6}$$

Independent of the values of ω_p, the above process is called *SOR process* even though this nomenclature is sometimes reserved for the case $\omega_p > 1$. Now, a large variety of combined methods can be constructed depending on the rootfinding method for solving (2.5), as well as the number of steps required to this end. Thus, for example, one can obtain the *exact nonlinear SOR* or *m-step SOR–Newton process* [13] and so on. Now, if the Jacobian of F, denoted by $F'(x)$, at the solution x^* of system (2.4) is an $\mathscr{M}$-matrix [21, 13], the iterates of all the above-mentioned methods will converge to x^* provided that $\omega_p \in (0, 1]$ [13].

(c) *A Modified Bisection Method*

It is well known that a root of $f(x) = 0$, where the function $f\colon [a, b] \subset \mathbb{R} \to \mathbb{R}$ is continuous, is guaranteed to exist in the interval $[a, b]$ if the following criterion is fulfilled:

$$f(a) f(b) \leq 0. \tag{2.7}$$

This criterion is known as Bolzano's existence criterion and can be generalized to higher dimensions [24]. Based on this criterion various rootfinding methods, for example, bisection methods, are created. Here we shall use the bisection method which has been modified to the following simplified version described in [22, 23]. There it is reported that, in order to compute a root of $f(x) = 0$ where $f\colon [a, b] \subset \mathbb{R} \to \mathbb{R}$ is continuous, a simplified version of the bisection method leads to the iterative formula

$$x^{p+1} = x^p + \operatorname{sgn} f(x^0) \operatorname{sgn} f(x^p) h / 2^{p+1}, \qquad p = 0, 1, \ldots, \tag{2.8}$$

with $x^0 = a$, $h = b - a$ and where sgn defines the well-known sign function with values

$$\operatorname{sgn} \psi = \begin{cases} -1, & \text{if } \psi < 0; \\ 0, & \text{if } \psi = 0; \\ 1, & \text{if } \psi > 0. \end{cases} \tag{2.9}$$

Of course the iterations (2.8) converge to a root $r \in (a, b)$ if for some x^p, $p = 1, 2, \ldots,$ the following holds:

$$\operatorname{sgn} f(x^0) \operatorname{sgn} f(x^p) = -1. \tag{2.10}$$

Also, the number of iterations ν, which are required in obtaining an approximate root r^* such that $|r - r^*| \leq \delta$ for some $\delta \in (0, 1)$, is given by

$$\nu = \left\lceil \log_2 (h \delta^{-1}) \right\rceil, \tag{2.11}$$

where the notation $\lceil \cdot \rceil$ refers to the smallest integer not less than the real number quoted.

Instead of the iterative formula (2.8) we can also use

$$x^{p+1} = x^p - \operatorname{sgn} f(x^0) \operatorname{sgn} f(x^p) h/2^{p+1}, \qquad p = 0, 1, \ldots, \tag{2.12}$$

with $x^0 = b$ and $h = b - a$.

It is evident from (2.8) and (2.12) that the only computable information required by the bisection method is the algebraic signs of the function f. The one-dimensional rootfinding portion of our method employs the above modified bisection method. Alternatively, any one of the well-known one-dimensional methods [13, 6] can be used. We use bisection method since it is a global convergence method, it always converges within the given interval, it is optimal [19, 20], i.e., it possesses asymptotically the best rate of convergence, it has a known behavior concerning the number of iterations required when we seek a root with a predetermined accuracy, and last, but not least, it is the only method that can be applied to problems with imprecise function values.

3. THE NEW METHOD AND ITS CONVERGENCE

In order to find a point x^* which minimizes a given function $f\colon \mathscr{D} \subset \mathbb{R}^n \to \mathbb{R}$ in a specific domain $\mathscr{D}$, we shall try to obtain a sequence of points x^k, $k = 0, 1, \ldots$ which converges to x^*. So, using an arbitrary chosen starting point $x^0 = (x_1^0, x_2^0, \ldots, x_n^0) \in \mathscr{D}$, we solve the one-dimensional equation

$$f\left(x_1, x_2^0, \ldots, x_n^0\right) - f\left(x_1^0, x_2^0, \ldots, x_n^0\right) = 0, \tag{3.1}$$

for x_1 keeping all the other coordinates in their constant values. Now, if $\hat{x}_1$ is the solution of the above equation, then, of course, the point $(\hat{x}_1, x_2^0, \ldots, x_n^0)$ possesses the same function value with the point x^0, so it belongs to the contour line of x^0. Under the assumption of positive definitiveness of the Hessian matrix of f at x^* (which corresponds to the geometric concept of strict local convexity, which means that f curves up from x^* in all directions), we can claim that any point which belongs to the line segment with endpoints x^0 and $(\hat{x}_1, x_2^0, \ldots, x_n^0)$ possesses smaller function value than these endpoints. With this fact in mind we can now choose such a point, say for example

$$x_1^1 = x_1^0 + \gamma\left(\hat{x}_1 - x_1^0\right), \qquad \gamma \in (0,1) \tag{3.2}$$

and solve the one-dimensional equation

$$f\left(x_1^1, x_2, x_3^0, \dots, x_n^0\right) - f\left(x_1^1, x_2^0, x_3^0, \dots, x_n^0\right) = 0 \tag{3.3}$$

for x_2 keeping all the other coordinates in their constant values. Suppose now that $\hat{x}_2$ is the solution of Eq. (3.3). We can obtain a better approximation for the second component by taking

$$x_2^1 = x_2^0 + \gamma\left(\hat{x}_2 - x_2^0\right), \qquad \gamma \in (0,1). \tag{3.4}$$

We continue this process for the remaining coordinates to obtain the point $x^1 = (x_1^1, \dots, x_n^1)$. Now, replacing the starting point x^0 by x^1, we can repeat the above process to compute x^2 and so on until the final estimated point is computed according to a predetermined accuracy. For a geometric interpretation of the new method for $\gamma = \frac{1}{2}$ illustrated on the well-known Broyden's banded function [12] for $n = 2$, see Fig. 1.

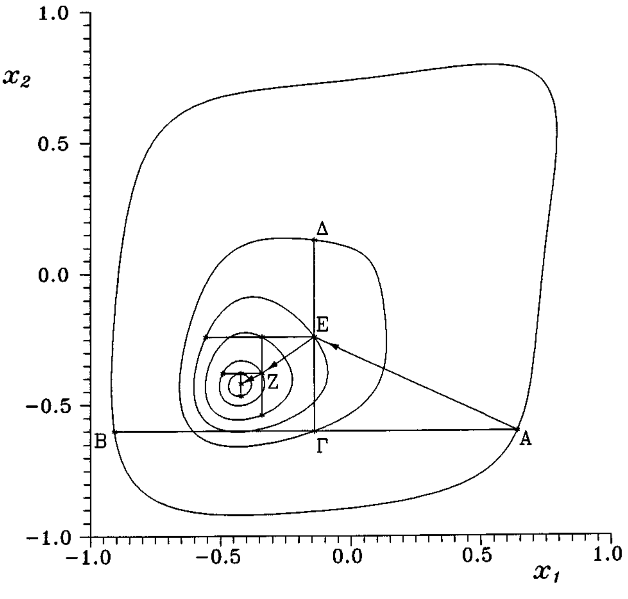

FIG. 1. A geometric interpretation of the new method for $\gamma = \frac{1}{2}$ is illustrated on the well-known Broyden's banded function [12] for $n = 2$ where $A = (x_1^0, x_2^0)$, $B = (\hat{x}_1, x_2^0)$, $\Gamma = (x_1^1, x_2^0)$, with $x_1^1 = (\hat{x}_1 + x_1^0)/2$, $\Delta = (x_1^1, \hat{x}_2)$, $E = (x_1^1, x_2^1)$, with $x_2^1 = (\hat{x}_2 + x_2^0)/2$, and $Z = (x_1^2, x_2^2)$.

To solve the above-mentioned one-dimensional equations we employ the iterative scheme (2.8) or (2.12). Alternatively, any one of the one-dimensional rootfinding methods (see [13, 17], etc.) can be used. We use this modified bisection method for the reasons explained in Section 2.

Of course, utilizing the above process we are able to obtain also local maxima. But if, in applying the previous one-dimensional bisection, we choose the endpoints a and b so that at the left endpoint a the ist component of the gradient has negative value or at the right endpoint b the ist component of the gradient has positive value, then clearly the new method deals with a local minimum. This can be easily handled by applying the iterative scheme (2.8) and taking the endpoint a and b from the relations

$$a = x_i^k - \tfrac{1}{2}\{1 + \operatorname{sgn} \partial_i f(x^k)\}h_i \qquad \text{and} \qquad b = a + h_i, \tag{3.5}$$

where ∂_i denotes the ist component of the gradient and h_i indicates the corresponding step size.

A question may be raised concerning the existence of the roots of Eq. (2.5), (i.e., the existence of the points $\hat{x}_i$), especially when the starting point is not close to a minimum. Of course, this existence can be detected by checking if the iterative scheme (2.8) converges to the right endpoint b, or the signs of the corresponding function values at the endpoints a and b are the same. When this happens we can apply Armijo's method for a few steps and then try again with our method. Our experience is that in many cases as well as for all the problems studied in this paper (see below in Section 4) the application of Armijo's method is not necessary. We have merged it in our algorithm for completeness.

With the above discussion in mind we can now state our algorithm where f indicates the objective function, $x^0 = (x_1^0, \dots, x_n^0)$ the starting point, $h = (h_1, \dots, h_n)$ the starting step sizes in each coordinate direction, γ and ζ relaxation coefficients, *MIT* the maximum number of iterations required, *MAR* the maximum number of Armijo's iterations required, and δ, ε the predetermined desired accuracies.

ALGORITHM 3.1. *Computation of an Unconstrained Local Minimum.*

Step 1 *Input*$\{f; x^0; h; \gamma; \zeta; MIT; MAR; \delta; \varepsilon\}$.

Step 2 Set $k = -1$.

Step 3 Set $i = 0$.

Step 4 If $k > MIT$, replace k by $k + 1$ and go to next step; otherwise, go to Step 16.

Step 5 Replace i by $i + 1$ and continue.

Step 6 Set $a = x_i^k - \frac{1}{2}\{1 + \operatorname{sgn} \partial_i f(x^k)\}h_i$ and $b = a + h_i$.

Step 7 Compute an approximate solution $\hat{x}$ of

$$f\left(x_1^{k+1},\dots,x_{i-1}^{k+1},x,x_{i+1}^k,\dots,x_n^k\right) - f\left(x_1^{k+1},\dots,x_{i-1}^{k+1},x_i^k,x_{i+1}^k,\dots,x_n^k\right) = 0$$

by applying the iterative scheme (2.8) in (a,b) within accuracy δ.

Step 8 If $\hat{x} \geq b - \delta$, set $y^0 = (x_1^{k+1},\dots,x_{i-1}^{k+1},x_i^k,x_{i+1}^k,\dots,x_n^k)$ and go to Step 14; otherwise continue.

Step 9 Set $x_i^{k+1} = x_i^k + \gamma(\hat{x} - x_i^k)$.

Step 10 If $i < n$, go to Step 5.

Step 11 If $\|x^{k+1} - x^k\| \leq \varepsilon$, go to Step 16.

Step 12 Set $x^{k+1} = x^k + \zeta(x^{k+1} - x^k)$.

Step 13 If $\operatorname{sgn}(f(x^{k+1}) - f(x^k)) \leq 0$, return to Step 3; otherwise set $y^0 = x^k$ and continue.

Step 14 Apply Armijo's algorithm utilizing the starting value y^0 and take its output value y^{MAR}.

Step 15 Set $x^{k+1} = y^{MAR}$ and return to Step 3.

Step 16 *Output* $\{x^k; f(x^k); \nabla f(x^k)\}$.

When the values of the objective function or the gradient values can be accurately obtained we may also use the following convergence criteria (see [18]) at Step 11 of our algorithm:

(i) $$\left|\frac{f(x^{k+1}) - f(x^k)}{f(x^k)}\right| \leq \varepsilon,$$

(ii) $$\|\nabla f(x^{k+1})\| \leq \varepsilon.$$

Note that here γ and ζ are chosen at the beginning (for example $\gamma = \frac{1}{2}$ and $\zeta = 1$), but one may utilize different relaxation coefficients γ_i^k and ζ_i^k in each iteration and for each component if required.

Furthermore, when the gradient values can be accurately obtained, an estimate of the relaxation parameter γ at Step 9 of the algorithm may be achieved using for example the ist coordinate of the intersection point of the lines specified by the points $\hat{x}$ and x_i^k and the corresponding values of the ist component of the gradient. In this case the coefficient γ can be obtained so that the point x_i^{k+1} at Step 9 of the algorithm coincides with

$$x_i^{k+1} = \frac{x_i^k \, \partial_i f(p_1) - \hat{x} \, \partial_i f(p_2)}{\partial_i f(p_1) - \partial_i f(p_2)}, \tag{3.6}$$

with $p_1 = (x_1^{k+1}, \dots, x_{i-1}^{k+1}, x_i^k, x_{i+1}^k, \dots, x_n^k)$ and $p_2 = (x_1^{k+1}, \dots, x_{i-1}^{k+1}, \hat{x}, x_{i+1}^k, \dots, x_n^k)$. Thus, in this case, we are able to get an estimation of γ at each iteration and for each coordinate using the following relation:

$$\gamma_i^k = \frac{\partial_i f(p_2)}{\partial_i f(p_2) - \partial_i f(p_1)}. \tag{3.7}$$

Similarly, an estimate of the relaxation parameter ζ at Step 12 of the algorithm can be achieved using the values of the ist component of the gradient at the points x^k and x^{k+1} and minimizing the produced one-dimensional quadratic function. In this case an estimate of the coefficient ζ may be obtained so that the point x_i^{k+1} at Step 12 of the algorithm coincides with the following one:

$$x_i^{k+1} = \frac{x_i^{k+1} \, \partial_i f(x^k) - x_i^k \, \partial_i f(x^{k+1})}{\partial_i f(x^k) - \partial_i f(x^{k+1})}. \tag{3.8}$$

Thus, in this case, we can have an estimation of ζ at each iteration and for each coordinate using the following relation:

$$\zeta_i^k = \frac{\partial_i f(x^k)}{\partial_i f(x^k) - \partial_i f(x^{k+1})}. \tag{3.9}$$

In order to give a convergence result for our method the following nonlinear SOR theorem [13] is needed:

THEOREM 3.1. *Suppose that* $F = (f_1, \dots, f_n): \mathscr{D} \subset \mathbb{R}^n \to \mathbb{R}^n$ *is continuously differentiable on an open neighborhood* $\mathscr{S}_0 \subset \mathscr{D}$ *of a point* $x^* \in \mathscr{D}$ *for which* $F(x^*) = \Theta^n$. *Consider the decomposition of the Jacobian* $F'(x)$ *into its diagonal, strictly lower-triangular and strictly upper-triangular parts:*

$$F'(x) = D(x) - L(x) - U(x). \tag{3.10}$$

Suppose further that $D(x^*)$ *is nonsingular and* $\rho(\Phi_\omega(x^*)) < 1$, *where* $\rho(A)$ *indicates the spectral radius of* A *and where* $\Phi_\omega(x)$ *is defined by*

$$\Phi_\omega(x) = [D(x) - \omega L(x)]^{-1}[(1-\omega)D(x) + \omega U(x)], \tag{3.11}$$

with $\omega > 0$. *Then there exists an open ball* $\mathscr{S} = \mathscr{S}(x^*, r)$ *in* $\mathscr{S}_0$, (*where* $\mathscr{S}(x^*, r)$ *denotes the open ball centered at* x^* *with radius* r), *such that, for any* $x^0 \in \mathscr{S}$, *there is a unique sequence* $\{x^k\} \subset \mathscr{S}$ *which satisfies the nonlinear*

SOR prescription such that $\lim_{k\to\infty} x^k = x^*$.

Proof. See [13], p. 326. ▌

A direct corollary of the above result is the following:

COROLLARY 3.1. *Suppose that* $F = (f_1, \dots, f_n)\colon \mathcal{D} \subset \mathbb{R}^n \to \mathbb{R}^n$ *is continuously differentiable on an open neighborhood* $\mathcal{S}_0 \subset \mathcal{D}$ *of a point* $x^* \in \mathcal{D}$ *for which* $F(x^*) = \Theta^n$. *Assume, further, that the Jacobian* $F'(x^*)$ *is symmetric and positive definite. Then there exists an open ball* $\mathcal{S} = \mathcal{S}(x^*, r)$ *in* $\mathcal{S}_0$ *such that* x^* *is a point of attraction of the nonlinear SOR scheme, for any* $x^0 \in \mathcal{S}$ *and for any* $\omega \in (0, 2)$.

Proof. Consider the decomposition (3.10) of $F'(x)$. Since $F'(x^*)$ is symmetric and positive definite, then $D(x^*)$ is nonsingular (see [21], p. 80). Now, by virtue of Ostrowski Theorem (see [21], p. 77), $\rho(\Phi_\omega(x^*)) < 1$ for any $\omega \in (0, 2)$. Thus, by Theorem 3.1, the corollary is proved. ▌

Next, we give our convergence result.

THEOREM 3.2. *Let* $f\colon \mathbb{R}^n \to \mathbb{R}$ *be twice continuously differentiable on an open neighborhood* $\mathcal{S}_0 \subset \mathcal{D}$ *of a point* $x^* \in \mathcal{D}$ *for which* $\nabla f(x^*) = \Theta^n$ *and* $\nabla^2 f(x^*)$ *is positive definite. Then there exists an open ball* $\mathcal{S} = \mathcal{S}(x^*, r)$ *in* $\mathcal{S}_0$ *such that the sequence* $\{x^k\}$ *generated by the new scheme converges to the point* x^* *which minimizes* f.

Proof. Clearly, the necessary and sufficient conditions for the point x^* to be a local minimizer of the function f are satisfied by the hypothesis $\nabla f(x^*) = \Theta$ and the assumption of positive definitiveness of the Hessian at x^* (see for example [13, 6]). Of course, finding such a point is equivalent to obtaining the corresponding solution $x^* \in \mathcal{D}$ of the equation

$$\nabla f(x) = \Theta^n, \tag{3.12}$$

or equivalently to solving the following system of equations:

$$\begin{aligned} \partial_1 f(x_1, x_2, \dots, x_n) &= 0, \\ \partial_2 f(x_1, x_2, \dots, x_n) &= 0, \\ &\vdots \\ \partial_n f(x_1, x_2, \dots, x_n) &= 0. \end{aligned} \tag{3.13}$$

Now, applying the nonlinear SOR method (2.5) to the above system we have to find at the $(k+1)$st iteration a solution $\hat{x}_i$ of the one-dimensional equation

$$\partial_i f\left(x_1^{k+1}, \dots, x_{i-1}^{k+1}, x, x_{i+1}^{k}, \dots, x_n^{k}\right) = 0, \tag{3.14}$$

and set

$$x_i^{k+1} = x_i^k + \omega\left(\hat{x}_i - x_i^k\right), \qquad \omega \in (0,2). \tag{3.15}$$

Evidently, the sequence of the above points x^k can be generated by the new method for some values of γ and ζ. Now, since $\nabla^2 f(x^*)$ is symmetric and positive definite, then, by Corollary 3.1, x^* is a point of attraction of (3.15). Thus the theorem is proved. ▮

4. NUMERICAL APPLICATIONS

The algorithm described in Section 3 has been implemented using a new FORTRAN program OPTBIS (OPTimization BISection). OPTBIS was tested on the University of Patras HP-715 system as well as on an Express IBM PC compatible with random problems of varying dimensions. Our experience is that the algorithm behaved predictably and reliably and the results were quite satisfactory. Some typical computational results are given below. For the following problems, the reported parameters are:

— n dimension,

— $x^0 = (x_1, x_2, \dots, x_n)$ starting point,

— $h = (h_1, h_2, \dots, h_n)$ starting stepsizes in each coordinate direction,

— $x^* = (x_1^*, x_2^*, \dots, x_n^*)$ approximate local minimum computed within an accuracy of $\varepsilon = 10^{-8}$,

— IT the total number of iterations required in obtaining x^*,

— FE the total number of function evaluations,

— ASF the total number of function algebraic signs that are required for applying the iterative schemes (2.8) and

— ASG the total number of algebraic signs of the components of the gradient.

In Tables I–XII we compare the numerical results obtained, for various starting points, by applying steepest descent methods as well as conjugate gradient methods and variable metric methods, with the corresponding numerical results of the method presented in this paper obtained on the University of Patras HP-715 system. In these tables x^0 indicates the starting point while the index α indicates the classical starting point.

Furthermore, D indicates divergence or nonconvergence after 50000 iterations while FR, PR, and BFGS indicate the corresponding results obtained by the Fletcher–Reeves [8, 17], Polak–Ribiere [16, 17], and Broyden–Fletcher–Goldfarb–Shanno [17] algorithms, respectively. Armijo's algorithm was applied utilizing the starting values of $\eta_0 = 1$.

EXAMPLE 4.1. *Quadratic function.* This example gives the local minimum for the function

$$f(x_1, x_2, \dots, x_n) = x_1^2 + x_2^2 + \cdots + x_n^2 - r, \tag{4.1}$$

with $f(x^*) = -r$ at $x^* = (0, \dots, 0)$. As our starting values we utilized $n = 4$, $r = 100$, $x^0 = (99.99, 99.99, 99.99, 99.99)$, and $h = (200, 200, 200, 200)$. We obtained the $x^* = (10^{-8}, 10^{-8}, 10^{-8}, 10^{-8})$ after IT = 1, ASF = 29, ASG = 4. By changing x^0 to $x^0 = (-1, 1, 1, -1)$ and h to $h = (2, 2, 2, 2)$, we obtained the $x^* = (10^{-8}, 10^{-8}, 10^{-8}, 10^{-8})$ after IT = 1, ASF = 29, ASG = 4. (See Table I.)

EXAMPLE 4.2. *Olympus function.* In this case we obtain the local minimum for the function

$$f(x_1, x_2) = J_1^2(x_1) + J_1^2(x_2), \tag{4.2}$$

where J_1 indicate the first-order Bessel function [17], where

$$J_1(x) = \sum_{k=0}^{\infty} (-1)^k \frac{(x/2)^{2k+1}}{k!(k+1)!}. \tag{4.3}$$

Here we have $f(x^*) = 0$ at $x^* = (0, 0)$. For this example TFE indicates the total number of the terms $(x/2)^{2k+1}/(k!(k+1)!)$ of Eq. (4.3) required in obtaining the function values of Eq. (4.2) and the corresponding gradient values to the predetermined accuracy. As our starting values we utilized $x^0 = (-5, -5)$ and $h = (8, 8)$. We obtained the $x^* = (10^{-8}, 10^{-8})$ after IT = 3, TFE = 51. By changing x^0 to $x^0 = (100, -100)$ and h to $h =$

TABLE I
Quadratic Function

	Armijo		FR		PR		BFGS		OPTBIS		
x^0	IT	FE	IT	FE	IT	FE	IT	FE	IT	ASF	ASG
$(-1, 1, 1, -1)$	17	123	3	93	3	170	2	67	1	29	4
$(10, 20, 30, 40)$	22	158	3	92	3	100	2	70	1	29	4
$(-20, 40, -60, 80)$	23	165	3	93	3	99	2	69	1	29	4
$(-50, -25, -50, -25)$	22	158	3	93	3	171	2	69	1	29	4
$(80, -80, -80, 80)$	23	165	3	93	3	169	2	67	1	29	4
$(99.99, 99.99, 99.99, 99.99)$	23	165	3	93	3	169	2	67	1	29	4

TABLE II
Olympus Function

	Armijo		FR		PR		BFGS		OPTBIS	
x^0	IT	TFE	IT	TFE	IT	TFE	IT	TFE	IT	TFE
$(-5, -5)$	D	D	3	644	3	676	3	558	3	51
$(2, 2)$	29	153	3	961	3	995	3	1094	7	119
$(-2, 4)$	28	391	D	D	9	630	6	470	5	85
$(-10, 10)$	52	3964	3	2273	3	2489	3	2047	7	119
$(50, -20)$	D	D	D	D	D	D	D	D	6	102
$(100, -100)$	D	D	D	D	D	D	D	D	2	34

$(200, 200)$, we obtained the $x^* = (10^{-8}, 10^{-8})$ after IT = 2, TFE = 34. (See Table II.)

The advantage of applying our method to this kind of problem is obvious since, with this method, only the signs need to be computed which can be achieved by taking into account relatively few terms of (4.3).

EXAMPLE 4.3. *Watson function*, [12]. In this case f is given by

$$f(x) = \sum_{i=1}^{m} f_i^2(x), \qquad m = 31, \tag{4.4}$$

where

$$f_i(x) = \sum_{j=2}^{n} (j-1) x_j t_i^{j-2} - \left(\sum_{j=1}^{n} x_j t_i^{j-1} \right)^2 - 1,$$

$$1 \le i \le 29, \qquad t_i = i/29, \tag{4.5}$$

$$f_{30}(x) = x_1, \qquad f_{31}(x) = x_2 - x_1^2 - 1, \tag{4.6}$$

where, for $n = 2$, we have $f(x^*) = 0$ at $x^* = (-0.50136701, 1.07364983)$. As our starting values we utilized $x^0 = (0, 0)$ and $h = (2, 2)$. We obtained the $x^* = (-0.50136701, 1.07364983)$ after IT = 4, ASF = 60, ASG = 8. By changing x^0 to $x^0 = (-1, -1)$ and h to $h = (3, 3)$, we obtained the $x^* = (-0.50136701, 1.07364983)$ after IT = 3, ASF = 45, ASG = 6. (See Table III.)

EXAMPLE 4.4. *Brown badly scaled function*, [12]. In this example f is given by

$$f(x) = \left(x_1 - 10^6\right)^2 + \left(x_2 - 2 \cdot 10^{-6}\right)^2 + (x_1 x_2 - 2)^2, \tag{4.7}$$

TABLE III
Watson Function, $n = 2$.

	Armijo		FR		PR		BFGS		OPTBIS		
x^0	IT	FE	IT	FE	IT	FE	IT	FE	IT	ASF	ASG
$(0,0)^{\alpha}$	14	121	9	159	7	132	6	98	4	60	8
$(-1,-1)$	17	160	9	165	7	151	6	101	3	45	6
$(-5,1)$	21	205	12	214	8	146	7	114	3	45	6
$(2,-1)$	18	164	9	172	8	148	7	115	6	90	12
$(3,4)$	21	216	12	230	8	184	7	129	6	90	12
$(1000,1000)$	63	1036	17	388	11	262	10	195	5	75	10

with $f(x^*) = 0$ at $x^* = (10^6, 2 \cdot 10^{-6})$. As our starting values we utilized $x^0 = (1,1)$ and $h = (10^7, 10^3)$. We obtained the $x^* = (10^6, 2 \cdot 10^{-6})$ after IT = 5, ASF = 75, ASG = 10. By changing x^0 to $x^0 = (10^7, 1)$ and h to $h = (10^7, 10^3)$, we obtained the $x^* = (10^6, 2 \times 10^{-6})$ after IT = 2, ASF = 30, ASG = 4. (See Table IV.)

EXAMPLE 4.5. *Weber–Werner singular function* [26]. This example gives the local minimum for the function

$$f(x) = \left(x_1^2 - 2x_1 + \frac{x_2^3}{3} + \frac{2}{3}\right)^2 + \left(x_1^3 - x_1 x_2 - 2x_1 + \frac{x_2^2}{2} + \frac{3}{2}\right)^2, \tag{4.8}$$

with $f(x^*) = 0$ at $x^* = (1,1)$. As our starting values we utilized, $x^0 = (2,-1)$ and $h = (3,3)$. We obtained the $x^* = (1,1)$ after IT = 7, ASF = 105, ASG = 14. By changing x^0 to $x^0 = (1.1, 1.1)$ and h to $h = (2,2)$, we obtained the $x^* = (1,1)$ after IT = 2, ASF = 30, ASG = 4. (See Table V.)

TABLE IV
Brown Badly Scaled Function

	Armijo		FR		PR		BFGS		OPTBIS		
x^0	IT	FE	IT	FE	IT	FE	IT	FE	IT	ASF	ASG
$(1,1)^{\alpha}$	D	D	D	D	D	D	D	D	5	75	10
$(-1,1)$	D	D	D	D	D	D	D	D	5	75	10
$(2,2)$	D	D	D	D	D	D	D	D	5	75	10
$(10000,1)$	D	D	D	D	D	D	15	238	2	30	4
$(-1000,1000)$	D	D	D	D	D	D	25	405	6	90	12
$(10^7,1)$	D	D	D	D	D	D	D	D	2	30	4

TABLE V
Weber–Werner Singular Function

	Armijo		FR		PR		BFGS		OPTBIS		
x^0	IT	FE	IT	FE	IT	FE	IT	FE	IT	ASF	ASG
$(2, -1)^{\alpha}$	81	284	74	942	12	192	14	216	7	105	14
$(-1, 1)$	80	266	49	649	11	185	11	178	2	30	4
$(1.1, 1.1)$	69	217	41	537	12	182	12	178	2	30	4
$(-1, -1)$	80	262	52	695	17	257	17	266	5	75	10
$(2, 0)$	88	300	79	1008	12	194	10	164	2	30	4
$(-1000, 1.1)$	117	1370	D	D	D	D	D	D	3	45	6

EXAMPLE 4.6. *Kearfott function* [10]. In this case f is given by

$$f(x) = \left(x_1^2 + x_2^2 - 2\right)^2 + \left(x_1^2 - x_2^2 - 1\right)^2, \tag{4.9}$$

with $f(x^*) = 0$ at $x^* = (\pm\sqrt{1.5}, \pm\sqrt{0.5})$. As our starting values utilized $x^0 = (1, 1)$ and $h = (1, 1)$. We obtained the $x^* = (\sqrt{1.5}, \sqrt{0.5})$ after IT = 2, ASF = 30, ASG = 4. By changing x^0 to $x^0 = (-1, -1)$ and h to $h = (1, 1)$, we obtained the $x^* = (-\sqrt{1.5}, -\sqrt{0.5})$ after IT = 2, ASF = 30, ASG = 4. (See Table VI.)

EXAMPLE 4.7. *Broyden banded function* [12]. In this example f is given by

$$f(x) = \sum_{i=1}^{n} f_i^2(x), \tag{4.10}$$

TABLE VI
Kearfott Function

	Armijo		FR		PR		BFGS		OPTBIS		
x^0	IT	FE	IT	FE	IT	FE	IT	FE	IT	ASF	ASG
$(1, 1)$	9	72	10	138	7	101	6	82	2	30	4
$(1.2, 0.6)$	15	106	10	140	6	88	5	72	2	30	4
$(1, 0.5)$	17	120	7	104	6	93	5	76	2	30	4
$(-1, -1)$	9	72	10	138	7	101	6	82	2	30	4
$(1.5, 1.5)$	16	116	9	129	7	104	6	87	3	45	6
$(100, -1000)$	35	477	7	151	7	151	7	151	2	30	4

with

$$f_i(x) = x_i\left(2 + 5x_1^2\right) + 1 - \sum_{j \in J_i} x_j\left(1 + x_j\right), \tag{4.11}$$

and where

$$J_i = \{j\colon j \neq i, \max(1, i - m_l) \leq j \leq \min(n, i + m_u)\}$$
$$\text{and} \qquad m_l = 5, m_u = 1. \tag{4.12}$$

Here we have for $n = 2$, $f(x^*) = 0$ at $x^* = (-0.42730462, -0.42730462)$. As our starting values we utilized $x^0 = (-1, -1)$ and $h = (2, 2)$. We obtained the x^* after IT = 3, ASF = 45, ASG = 6. By changing x^0 to $x^0 = (-3, -4)$ and h to $h = (5, 5)$, we obtained the x^* after IT = 4, ASF = 60, ASG = 8.

For $n = 3$ we get $f(x^*) = 0$ at $x^* = (-0.42783984, -0.55240668, -0.47614734)$. As our starting values we utilized $x^0 = (-1, -1, -1)$ and $h = (2, 2, 2)$. We obtained the x^* after IT = 4, ASF = 88, ASG = 12. By changing x^0 to $x^0 = (0, 10^3, 0)$ and h to $h = (1100, 1100, 1100)$, we obtained the x^* after IT = 6, ASF = 132, ASG = 18. (See Tables VII and VIII.)

EXAMPLE 4.8. *Trigonometric function* [12]. In this case f is given by:

$$f(x) = \sum_{i=1}^{n} f_i^2(x), \tag{4.13}$$

where

$$f_i(x) = n - \sum_{j=1}^{n} \cos x_j + i(1 - \cos x_i) - \sin x_i. \tag{4.14}$$

TABLE VII
Broyden Banded Function, $n = 2$

	Armijo		FR		PR		BFGS		OPTBIS		
x^0	IT	FE	IT	FE	IT	FE	IT	FE	IT	ASF	ASG
$(-1, -1)^{\alpha}$	14	136	4	103	4	78	3	51	3	45	6
$(-1, -2)$	18	198	11	187	7	131	7	121	3	45	6
$(-3, -4)$	20	236	10	190	8	157	8	150	4	60	8
$(-1, -4)$	22	276	8	145	7	132	6	113	4	60	8
$(100, 200)$	42	933	D	D	D	D	D	D	4	60	8
$(1000, -1)$	53	1409	D	D	D	D	D	D	3	45	6

TABLE VIII
Broyden Banded Function, $n = 3$

x^0	Armijo		FR		PR		BFGS		OPTBIS		
	IT	FE	IT	FE	IT	FE	IT	FE	IT	ASF	ASG
$(-1, -1, -1)^\alpha$	15	162	11	183	8	137	7	120	4	88	12
$(10, -10, 10)$	32	468	335	5789	16	294	13	228	9	198	27
$(0, 10000, 0)$	62	1563	D	D	D	D	D	D	6	132	18
$(0, 0, 500)$	53	1298	D	D	D	D	D	D	10	220	30
$(-100, -200, -300)$	46	1093	D	D	D	D	D	D	9	198	27
$(-0.4, -0.5, 1000)$	55	1492	D	D	D	D	D	D	6	132	18

Here, for $n = 3$, we get $f(x^*) = 0$ at $x^* = (0, 0, 0)$, $x^* = (0.13865866, 0.15238123, 0.46778723)$ and $x^* = (0.22497195, 0.51431111, 0.30305234)$. As our starting values we utilized, $x^0 = (3^{-1}, 3^{-1}, 3^{-1})$ and $h = (1, 1, 1)$. We obtained the $x^* = (0, 0, 0)$ after IT = 5, ASF = 110, ASG = 15. By changing x^0 to $x^0 = (-0.25, -0.5, -0.75)$ and h to $h = (1, 1, 1)$, we obtained the $x^* = (0, 0, 0)$ after IT = 4, ASF = 88, ASG = 12. (See Table IX.)

EXAMPLE 4.9. *Linear function-rank 1* [12]. In this case f is given by

$$f(x) = \sum_{i=1}^{n} f_i^2(x), \tag{4.15}$$

where

$$f_i(x) = -1 + i \sum_{j=1}^{n} j x_j, \tag{4.16}$$

TABLE IX
Trigonometric Function, $n = 3$

x^0	Armijo		FR		PR		BFGS		OPTBIS		
	IT	FE	IT	FE	IT	FE	IT	FE	IT	ASF	ASG
$(3^{-1}, 3^{-1}, 3^{-1})^\alpha$	50	211	167	2652	13	199	11	168	5	110	15
$(1, 1, 1)$	41	188	184	2741	16	241	11	175	6	132	18
$(1, -1, 1)$	33	166	57	864	13	211	14	229	5	110	15
$(-0.25, -0.5, -0.75)$	14	93	14	209	7	112	8	127	4	88	12
$(-1, -1, -1)$	16	109	24	357	11	182	10	168	1	22	3
$(-2, 3, -4)$	39	191	12	182	12	182	10	160	5	110	15

with $f(x^*) = \frac{3}{7}$ at any point where $\sum_{j=1}^{n} jx_j^* = \frac{3}{7}$. As our starting values we utilized $n = 3$, $x^0 = (1, 1, 1)$ and $h = (2, 2, 2)$. We obtained the $x^* = (0.60204082,\ 0.20408163,\ -0.19387755)$ after IT = 2, ASF = 44, ASG = 6. By changing x^0 to $x^0 = (-1, -1, -1)$ and h to $h = (2, 2, 2)$, we obtained the $x^* = (0.60204082,\ 0.20408163,\ -0.19387755)$ after IT = 2, ASF = 44, ASG = 6. (See Table X.)

EXAMPLE 4.10. *Penalty function* I [12]. In this case f is given by

$$f(x) = \sum_{i=1}^{m} f_i^2(x), \qquad m = n + 1, \tag{4.17}$$

where

$$f_i(x) = a^{1/2}(x_i - 1), \qquad 1 \le i \le n, \qquad a = 10^{-5}, \tag{4.18}$$

$$f_{n+1}(x) = \left(\sum_{j=1}^{n} x_j^2 \right) - \frac{1}{4}. \tag{4.19}$$

Here, for $n = 4$, we have $f(x^*) = 10^{-5} \times 2.24997\ldots$ at $x^* = (-0.24998750, -0.24998750, -0.24998750, -0.24998750)$. As our starting values we utilized, $x^0 = (1, 2, 3, 4)$ and $h = (5, 5, 5, 5)$. We obtained the $x^* = (-0.24998750,\ -0.24998750,\ -0.24998750,\ -0.24998750)$ after IT = 1, ASF = 29, ASG = 4. By changing x^0 to $x^0 = (10, 20, 30, 40)$ and h to $h = (50, 50, 50, 50)$, we obtained the x^* after IT = 1, ASF = 29, ASG = 4. (See Table XI.)

EXAMPLE 4.11. *Botsaris function* [3]. In this case f is given by

$$f(x) = 100\left[x_n - \left(\frac{1}{n-1} \sum_{i=1}^{n-1} x_i \right)^2 \right]^8 + \sum_{i=1}^{n-1} (1 - x_i)^8, \tag{4.20}$$

TABLE X
Linear Function-Rank 1, $n = 3$

	Armijo		FR		PR		BFGS		OPTBIS		
x^0	IT	FE	IT	FE	IT	FE	IT	FE	IT	ASF	ASG
$(1, 1, 1)^\alpha$	12	156	3	68	3	155	2	54	2	44	6
$(-1, -1, -1)$	12	156	3	68	3	144	2	54	2	44	6
$(1, 2, 3)$	13	168	3	69	3	153	2	53	2	44	6
$(-1, 1, -1)$	12	155	3	77	3	127	2	46	1	22	3
$(10, 20, 30)$	14	182	3	75	3	146	2	46	2	44	6
$(1000, -1000, -1000)$	17	220	3	61	3	112	2	40	1	22	3

TABLE XI
Penalty Function I, $n = 4$

	Armijo		FR		PR		BFGS		OPTBIS		
x^0	IT	FE	IT	FE	IT	FE	IT	FE	IT	ASF	ASG
$(1, 2, 3, 4)^\alpha$	17	150	2017	43123	82	1491	84	1468	1	29	4
$(3, 3, 3, 3)$	19	165	4	96	4	98	3	71	1	29	4
$(2, 4, 6, 8)$	19	181	30	540	22	404	21	384	1	29	4
$(10, 20, 30, 40)$	23	255	30	541	19	360	18	348	1	29	4
$(-100, 100, -100, 100)$	27	345	194	3657	55	991	57	996	1	29	4
$(100, 200, 300, 400)$	30	423	2015	43093	81	1478	3	86	1	29	4

with $f(x^*) = 0$ at $x^* = (1, 1, \ldots, 1)$. As our starting values we utilized $n = 10$, $x^0 = (-1.2, -1, \ldots, -1.2, -1)$, and $h = (4, \ldots, 4)$. We obtained the $x^* = (1, 1, \ldots, 1)$ after IT = 4, ASF = 284, ASG = 40. By changing x^0 to $x^0 = (-6, \ldots, -6)$ and h to $h = (10, \ldots, 10)$, we obtained the $x^* = (1, 1, \ldots, 1)$ after IT = 5, ASF = 355, ASG = 50. (See Table XII.)

5. CONCLUDING REMARKS AND FURTHER RESEARCH

This paper describes a new efficient numerical method for computing an unconstrained local minimum of functions $f\colon \mathscr{D} \subset \mathbb{R}^n \to \mathbb{R}$. This method always converges to a local minimum of f in $\mathscr{D}$ provided that the assumptions for the existence of such a minimum are fulfilled. The new method converges in one iteration on quadratic functions of n variables, it rapidly minimizes general functions and it is superior to other well-known optimization methods on a variety of classical test functions.

Our method avoids all information regarding the Hessian matrix as well as it avoids matrix inversions. It requires only that the algebraic signs of

TABLE XII
Botsaris Function, $n = 10$

	Armijo		FR		PR		BFGS		OPTBIS		
x^0	IT	FE	IT	FE	IT	FE	IT	FE	IT	ASF	ASG
$(-1.2, -1, \ldots, -1.2, -1)^\alpha$	1064	11854	10	334	10	331	13	359	4	284	40
$(4, 4, 4, 4, 4, 3, 3, 3, 3, 3)$	1284	14549	14	506	17	696	119	2851	5	355	50
$(-2, \ldots, -2)$	1055	11835	13	439	12	412	18	498	4	284	40
$(-3, 3, \ldots, -3, 3)$	866	9709	12	418	13	454	49	1199	5	355	50
$(-5, -4, \ldots, -5, -4)$	1066	12257	D	D	D	D	D	D	5	355	50
$(-6, \ldots, -6)$	1068	12405	D	D	D	D	D	D	5	355	50

the function and gradient values be correct, so that it can be applied to problems with imprecise function values.

Of course, having some previous knowledge about the optimal values of the relaxation parameters γ and ζ would significantly improve our method. We hope to address this approach in a future work.

It seems that the new method is not affected when the objective function has more than one extrema in $\mathscr{D}$ and it does not require the starting estimate of an extremum to be close to it.

Furthermore, preliminary investigations suggest that our method can be efficiently applied for the computation of the unconstrained global extrema (minimum and/or maximum) in some specific domain $\mathscr{G}$ by computing all the extrema in $\mathscr{G}$. The exact number of all the extrema in a bounded domain $\mathscr{G}$ can be obtained using the value of topological degree of $\nabla f(x)$ at the origin relative to $\mathscr{G}$, calculated on a Picard's extension [14, 15, 9]. An analysis and results for these approaches will appear in a subsequent communication [25].

ACKNOWLEDGMENT

The authors thank Professor C. A. Botsaris for useful discussions.

REFERENCES

1. L. Armijo, Minimization of functions having Lipschitz continuous first partial derivatives, *Pacific J. Math.* **16** (1966), 1–3.
2. M. Avriel, "Nonlinear Programming, Analysis and Methods," Prentice-Hall, Englewood Cliffs, NJ, 1976.
3. C. A. Botsaris, A curvilinear optimization method based upon iterative estimation of the eigensystem of the Hessian matrix, *J. Math. Anal. Appl.* **63** (1978), 396–411.
4. A. Cauchy, Méthode générale pour la résolution des systèmes d'équations simultanées, *Comp. Rend. Acad. Sci. Paris* **25** (1847), 536–538.
5. W. C. Davidon, Variable metric methods for minimizing, A.E.C., Research and Development Report, No. ANL-5990, Argonne National Laboratory, Argonne, IL, 1959.
6. J. E. Dennis Jr. and R. B. Schnabel, "Numerical Methods for Unconstrained Optimization and Nonlinear Equations," Prentice-Hall, Englewood Cliffs, NJ, 1983.
7. R. Fletcher and M. Powell, A rapidly convergent descent method for minimization, *Comput. J.* **6** (1963), 163–168.
8. R. Fletcher and C. Reeves, Function minimization by conjugate gradients, *Comput. J.* **7** (1964), 149–154.
9. B. J. Hoenders and C. H. Slump, On the calculation of the exact number of zeroes of a set of equations, *Computing* **30** (1983), 137–147.
10. B. Kearfott, An efficient degree-computation method for a generalized method of bisection, *Numer. Math.* **32** (1979), 109–127.

11. D. G. Luenberger, "Introduction to Linear and Nonlinear Programming," Addison-Wesley, Reading, MA, 1973.
12. B. J. Moré, B. S. Garbow, and K. E. Hillstrom, Testing unconstrained optimization, *ACM Trans. Math. Software* **7** (1981), 17–41.
13. J. M. Ortega and W. C. Rheinboldt, "Iterative Solution of Nonlinear Equations in Several Variables," Academic Press, New York, 1970.
14. E. Picard, Sur le notre des raciness communes à plushier équations simultanées, *J. Math. Pure. Appl. (4^e série)* **8** (1892), 5–24.
15. E. Picard, "Traité d'analyse," 3rd ed., Chap. 4.7, Gauthier-Villars, Paris, 1922.
16. E. Polak, "Computational Methods in Optimization," Academic Press, New York, 1971.
17. W. H. Press, B. P. Flannery, S. A. Teukolsky, and W. T. Vetterling, "Numerical Recipes, The Art of Scientific Computing," 2nd ed., Cambridge Univ. Press, New York, 1992.
18. S. S. Rao, "Optimization, Theory and Applications," Wiley Eastern Limited, New Delhi, 1984.
19. K. Sikorski, Bisection is optimal, *Numer. Math.* **40** (1982), 111–117.
20. K. Sikorski and G. M. Trojan, Asymptotic near optimality of the bisection method, *Numer. Math.* **57** (1990), 421–433.
21. R. Varga, "Matrix Iterative Analysis," Prentice-Hall, Englewood Cliffs, NJ, 1962.
22. M. N. Vrahatis, Solving systems of nonlinear equations using the nonzero value of the topological degree, *ACM Trans. Math. Software* **14** (1988), 312–329.
23. M. N. Vrahatis, CHABIS: A mathematical software package for locating and evaluating roots of systems of nonlinear equations, *ACM Trans. Math. Software* **14** (1988), 330–336.
24. M. N. Vrahatis, A short proof and a generalization of Miranda's existence theorem, *Proc. Amer. Math. Soc.* **107** (1989), 701–703.
25. M. N. Vrahatis, G. S. Androulakis, and G. E. Manoussakis, Locating and computing all the extrema (working title), manuscript in preparation.
26. H. Weber and W. Werner, On the accurate determination of nonisolated solutions of nonlinear equations, *Computing* **26** (1981), 315–326.