A dimension-reducing method for unconstrained optimization

T.N. Grapsa*, M.N. Vrahatis

Department of Mathematics, University of Patras, GR-261 10 Patras, Greece
Received 29 July 1994; revised 8 June 1995

Abstract

A new method for unconstrained optimization in $\mathbb{R}^n$ is presented. This method reduces the dimension of the problem in such a way that it can lead to an iterative approximate formula for the computation of $(n-1)$ components of the optimum while its remaining component is computed separately using the final approximations of the other components. It converges quadratically to a local optimum and it requires storage of order $(n-1) \times (n-1)$. Besides, it does not require a good initial guess for one component of the optimum and it does not directly perform gradient evaluations; thus it can be applied to problems with imprecise gradient values.

Moreover, a procedure for transforming the matrix formed by our method into a symmetric as well as into a diagonal one is presented. Furthermore, the proposed dimension-reducing scheme using finite difference gradient and Hessian is presented.

The methods have been implemented and tested. Performance information for well-known test functions is reported.

Keywords: Dimension-reducing method; Unconstrained optimization; Reduction to one-dimensional equations; Bisection method; Imprecise gradient values; Quadratic convergence

AMS classification: 65K10; 49M37; 90C30; 65H10

1. Introduction

There are a large variety of methods for unconstrained optimization of functions,

$$f: \mathcal{D} \subset \mathbb{R}^n \to \mathbb{R},$$

which require precise function and gradient values. However, in many optimization problems of practical interest the values of the objective functions and the corresponding gradients 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 [11].

* Corresponding author. E-mail: grapsa@math.upatras.gr.
Recently, new methods have been proposed [6–9] for the numerical solution of a system of nonlinear algebraic and/or transcendental equations:

\[ F(x) = \Theta^n = (0, 0, \ldots, 0), \]  

(1.2)

where \( F = (f_1, \ldots, f_n) : \mathcal{D} \subset \mathbb{R}^n \rightarrow \mathbb{R}^n \) is a continuously differentiable mapping in an open neighborhood \( \mathcal{D}^* \subset \mathcal{D} \) of a solution \( x^* \in \mathcal{D} \) of system (1.2). These methods incorporate the advantages of Newton and nonlinear SOR algorithms [14]. Specifically, although these methods use reduction to simpler one-dimensional nonlinear equations, they converge quadratically.

In this paper, we derive and apply a new iterative procedure for the computation of an unconstrained optimum of functions (1.1). This method is based on the methods studied in [6–9] and it incorporates the advantages of Newton and SOR algorithms. Although this new procedure uses reduction to simpler one-dimensional nonlinear equations, it generates a quadratically converging sequence of points in \( \mathbb{R}^{n-1} \) which converges to the \( n - 1 \) components of the optimum while the remaining component of the optimum is evaluated separately using the final approximations of the others. For this component an initial guess is not necessary and it is at the user’s disposal to choose which will be the remaining component, according to the problem. Also this method does not directly need any gradient evaluation and it compares favorably with quadratically convergent optimization methods.

In the next section the dimension-reducing optimization method is presented, its convergence is studied and the corresponding algorithm is proposed. In Section 3 we perturb the matrix obtained by the dimension-reducing procedure in order to transform it into a symmetric as well as into a diagonal one and we study the convergence of the produced scheme. In Section 4 we present the dimension-reducing optimization method utilizing finite difference approximations. Finally, we give numerical applications and some concluding remarks.

2. The dimension-reducing optimization method and its convergence

Notation. Throughout this paper \( \mathbb{R}^n \) is the \( n \)-dimensional real space of column vectors \( x \) with components \( x_1, x_2, \ldots, x_n \); \((y; z)\) represents the column vector with components \( y_1, y_2, \ldots, y_m, z_1, z_2, \ldots, z_k \); \( \partial_i f(x) \) denotes the partial derivative of \( f(x) \) with respect to the \( i \)-th variable \( x_i \); \( g(x) = (g_1(x), \ldots, g_n(x)) \) defines the gradient \( \nabla f(x) \) of the objective function \( f \) at \( x \) while \( H = [H_{ij}] \) defines the Hessian \( \nabla^2 f(x) \) of \( f \) at \( x \); \( \mathcal{A} \) denotes the closure of the set \( \mathcal{A} \) and \( f(x_1, \ldots, x_{i-1}, \cdot, x_{i+1}, \ldots, x_n) \) defines the mapping obtained by holding \( x_1, \ldots, x_{i-1}, x_{i+1}, \ldots, x_n \) fixed.

To obtain a sequence \( \{x^p\} \), \( p = 0, 1, \ldots, \) of points in \( \mathbb{R}^n \) which converges to a local optimum (critical) point \( x^* = (x_1^*, \ldots, x_n^*) \in \mathcal{D} \) of the function (1.1), we consider the sets \( \mathcal{B}_i, i = 1, \ldots, n, \) to be those connected components of \( g_i^{-1}(0) \) containing \( x^* \) on which \( \partial_ng_i \neq 0, \) for \( i = 1, \ldots, n \) respectively. Next, applying the Implicit Function Theorem [14] for each of the components \( g_i, i = 1, \ldots, n, \) we can find open neighborhoods \( \mathcal{A}_i^* \subset \mathbb{R}^{n-1} \) and \( \mathcal{A}_{2,i}^* \subset \mathbb{R}, i = 1, \ldots, n, \) of the points \( y^* = (x_1^*, \ldots, x_{n-1}^*) \) and \( x_n^* \) respectively, such that for any \( y = (x_1, \ldots, x_{n-1}) \in \mathcal{A}_i^* \) there exist
unique mappings $\varphi_i$ defined and continuous in $\mathcal{A}_1^*$ such that
\[ x_i = \varphi_i(y) \in \mathcal{A}_{2,i}, \quad i = 1, \ldots, n, \] (2.1)
and
\[ g_i(y; \varphi_i(y)) = 0, \quad i = 1, \ldots, n. \] (2.2)
Moreover, the partial derivatives $\partial_j \varphi_i, j = 1, \ldots, n - 1$, exist in $\mathcal{A}_1^*$ for each $\varphi_i, i = 1, \ldots, n$, they are continuous in $\mathcal{A}_1^*$ and they are given by
\[ \partial_j \varphi_i(y) = \frac{\partial_j g_i(y; \varphi_i(y))}{\partial_n g_i(y; \varphi_i(y))}, \quad i = 1, \ldots, n, \quad j = 1, \ldots, n - 1. \] (2.3)

Next, working exactly as in [7], we utilize Taylor's formula to expand the $\varphi_i(y), i = 1, \ldots, n$, about $y^p$. By straightforward calculations, we can obtain the following iterative scheme for the computation of the $n - 1$ components of $x^*$:
\[ y^{p+1} = y^p + A_p^{-1} V_p, \quad p = 0, 1, \ldots, \] (2.4)
where
\[ y^p = [x_i^p], \quad i = 1, \ldots, n - 1, \] (2.5)
\[ A_p = [a_{ij}] = \left[ \frac{\partial_j g_i(y^p; x_n^{p,i})}{\partial_n g_i(y^p; x_n^{p,n})} - \frac{\partial_j g_i(y^p; x_n^{p,n})}{\partial_n g_i(y^p; x_n^{p,i})} \right], \quad i, j = 1, \ldots, n - 1, \] (2.6)
and
\[ V_p = [v_i] = [x_n^{p,i} - x_n^{p,n}], \quad i = 1, \ldots, n - 1, \] (2.7)
with $x_n^{p,i} = \varphi_i(y^p)$. After a desired number of iterations of (2.4), say $p = m$, the $n$th component of $x^*$ can be approximated by means of the following relation:
\[ x_n^{m-1} = x_n^{m,n} - \sum_{j=1}^{n-1} \left( x_n^{m+1} - x_n^j \right) \frac{\partial_j g_i(y^m; x_n^{m,n})}{\partial_n g_i(y^m; x_n^{m,n})}. \] (2.8)

Next, we give a proof of convergence of the dimension-reducing optimization method (2.4).

**Theorem 2.1.** Suppose that the objective function $f: \mathcal{D} \subset \mathbb{R}^n \to \mathbb{R}$ is twice-continuously differentiable in an open neighborhood $\mathcal{D}^* \subset \mathcal{D}$ of a point $x^* = (x_1^*, \ldots, x_n^*) \in \mathcal{D}$ for which $g(x^*) = 0$. Let $\mathcal{B}_i, i = 1, \ldots, n$, be those connected components of $g_i^{-1}(0)$, containing $x^*$ on which $\partial_n g_i \neq 0$ for $i = 1, \ldots, n$ respectively. Then the iterations of (2.4) and the relationship (2.8) will converge to $x^*$ provided the Hessian of $f$ at $x^*$ is nonsingular and also provided the initial guess $y^0 = (x_n^0, \ldots, x_n^{0,n-1})$ is sufficiently close to $y^* = (x_1^*, \ldots, x_n^{n-1})$. Moreover the iterations $y^p, p = 0, 1, \ldots$, of (2.4) have order of convergence two.

**Proof.** The determinant of the matrix $A_*$, obtained from the matrix $A_p$ of (2.6) at $x^*$, can be written as follows:
\[ \det A_* = \frac{\det H(x^*)}{\prod_{i=1}^n \partial_n g_i(x^*)}. \] (2.9)
Now, since the Hessian of $f$ at $x^*$ is nonsingular then obviously $A_p$ is also nonsingular at $x^*$. 

The iterations (2.4) can be written as follows:
\[ y^{p+1} = y^p - A_p^{-1} M_p, \quad p = 0, 1, \ldots, \] (2.10)
where
\[ M_p = [m_i] = [-x_n^{p,i} + x_n^{p,n}], \quad i = 1, \ldots, n - 1. \] (2.11)
Consider now the mapping
\[ A = (\lambda_1, \ldots, \lambda_{n-1}): J_1 \cap \mathbb{R}^{n-1} \to \mathbb{R}^{n-1}, \quad \text{with} \]
\[ \lambda_i(y) = -\phi_i(y) + \phi_n(y), \quad i = 1, \ldots, n - 1, \] (2.12)
and denote the corresponding Jacobian by \( A' \); then the iterative scheme (2.4) is equivalent to the following one:
\[ y^{p+1} = y^p - A'(y^p)^{-1} A(y^p), \quad p = 0, 1, \ldots. \] (2.13)
Consequently, by the well-known Newton’s convergence theorem [13] for an initial guess \( y^0 \) sufficiently close to \( y^* \), the iterations \( y^p, \ p = 0, 1, \ldots \), of (2.4) converge to \( y^* \) and the order of convergence is two.
Suppose now that for some \( p \), for example \( p = m \), we obtain \( y^m = y^* \). Then, relation (2.8) yields
\[ x_n^{m+1} = \phi_n(y^*) = x_n^*. \] (2.14)
Thus the theorem is proved. \( \square \)

**Remark 2.2.** Relative procedures for obtaining \( x^* \) can be constructed by replacing \( x_n \) with any one of the components \( x_1, \ldots, x_{n-1} \), for example \( x_{int} \), and taking \( y = (x_1, \ldots, x_{int-1}, x_{int+1}, \ldots, x_n) \).

**Remark 2.3.** The above described method does not require the expressions \( \phi_i \) but only the values \( x_n^{p,i} \) which are given by the solution of the one-dimensional equations \( g_i(x_1^p, \ldots, x_{n-1}^p, \cdot) = 0 \). So, by holding \( y^p = (x_1^p, \ldots, x_{n-1}^p) \) fixed, we can solve the equations
\[ g_i(y^p; r_i^p) = 0, \quad i = 1, \ldots, n, \] (2.15)
for \( r_i^p \) in the interval \( (a, b) \) with an accuracy \( \delta \).

We can use any one of the well-known one-dimensional methods [14] to solve the above equations. Here we employ a modified bisection method described in [17, 18]. According to these, for the computation of a root of the equation \( \psi(x) = 0 \), where \( \psi: [a, b] \subset \mathbb{R} \to \mathbb{R} \) is continuous, we can use the bisection method which has been modified to the following version:
\[ x^{p+1} = x^p + \text{sgn} \psi(x^p) q/2^{p+1}, \quad p = 0, 1, \ldots, \] (2.16)
where \( x^0 = a, \ q = \text{sgn} \psi(a)(b - a) \) and where \( \text{sgn} \) defines the well-known sign function. This method computes with certainty a root when \( \text{sgn} \psi(x^0) \text{sgn} \psi(x^p) = -1 \) (see [19] for extensions).
It is evident from (2.16) that the only computable information required by the bisection method is the algebraic signs of the function \( \psi \). We use this method since it is a global convergence method, it always converges within the given interval and it is optimal [16], in the sense that it possesses asymptotically the best rate of convergence. Besides, the number of the iterations \( v \), which are
required to obtain an approximate root \( r^* \) such that \( |r - r^*| \leq \delta \) for some \( \delta \in (0, 1) \), is given by \( v = \lceil \log_2 ((b - a)\delta^{-1}) \rceil \), where the notation \( \lceil \cdot \rceil \) refers to the smallest integer not less than the real number quoted. Moreover, it is the only method that can be applied to problems with imprecise function values. This procedure has been efficiently implemented for the computation of all the zeros and extrema of a function [10].

The proposed method is illustrated in the following algorithm in pseudo-code where \( g = (g_1, g_2, \ldots, g_n) \) indicates the gradient of the objective function, \( x^0 \) the starting point, \( a = (a_1, a_2, \ldots, a_n) \), \( b = (b_1, b_2, \ldots, b_n) \) indicate the endpoints in each coordinate direction which are used for the one-dimensional bisection (2.16), \( \delta \) the predetermined accuracy for applying the procedure (2.16), MIT the maximum number of iterations required and \( \varepsilon_1, \varepsilon_2 \) the predetermined desired accuracies.

**Algorithm 1. Dimension-Reducing Optimization (DROPT)**

1. Input \( \{x^0; a; b; \delta; \text{MIT}; \varepsilon_1; \varepsilon_2\} \).
2. Set \( p = -1 \).
3. If \( p < \text{MIT} \) replace \( p \) by \( p + 1 \) and go to the next step; otherwise, go to Step 14.
4. If \( \|g(x^p)\| \leq \varepsilon_1 \) go to Step 14.
5. Find a coordinate int such that the following relation holds:

\[
\text{sgn} g_i(x_{i-1}^p, \ldots, x_{i-1}^p, a_{i-1}, a_{i-1}^p, \ldots, x_n^p) \cdot \text{sgn} g_i(x_{i-1}^p, \ldots, x_{i-1}^p, b_{i-1}, b_{i-1}^p, \ldots, x_n^p) = -1
\]

for all \( i = 1, 2, \ldots, n \). If this is impossible, apply Armijo's method (see below) and go to Step 4.
6. Compute the approximate solutions \( r_i \) for all \( i = 1, 2, \ldots, n \) of the equation

\[
g_i(x_{i-1}^p, \ldots, x_{i-1}^p, r_i, x_{i-1}^p, \ldots, x_n^p) = 0
\]

by applying the iterative scheme (2.16) in \( (a_{i-1}, b_{i-1}) \) within accuracy \( \delta \). Set \( x_{i-1}^p = r_i \).
7. Set \( y^p = (x_1^p, \ldots, x_{i-1}^p, x_{i-1}^p, x_{i+1}^p, \ldots, x_n^p) \).
8. Set the elements of the matrix \( A_p \) of relation (2.6) using \( x_{i-1} \) instead of \( x_n \).
9. Set the elements of the vector \( V_p \) of relation (2.7) using \( x_{i-1} \) instead of \( x_n \).
10. Solve the \((n-1) \times (n-1)\) linear system \( A_p s^p = -V_p \) for \( s^p \).
11. Set \( y^{p+1} = y^p + s^p \).
12. Compute \( x_{i-1} \) by virtue of relation (2.8) and set \( x^p = (y^p; x_{i-1}) \).
13. If \( \|s^p\| \leq \varepsilon_2 \) go to Step 14; otherwise return to Step 3.
14. Output \( \{x^p\} \).

The criterion in Step 5 ensures the existence of the solution \( r_i \), which will be computed at Step 6. If this criterion is not satisfied we apply Armijo's method [1, 20] for a few steps and then try again with our method. Our experience is that in many examples studied in various dimensions as well as for all the problems studied in this paper (see Section 5), the application of such a subprocedure is not necessary. We have merged it in our algorithm for completeness.

Based on this, we give the following subprocedure, where \( \text{MAR} \) is the maximum number of Armijo's iterations required, \( \eta \) is an arbitrary assigned positive number and \( \varepsilon \) the predetermined desired accuracy.
Algorithm 2. Armijo’s Modification of Steepest Descent
1. Input \( \{x^0; \text{MAR}; \eta_0; \varepsilon\} \).
2. Set \( k = -1 \).
3. If \( k < \text{MAR} \), replace \( k \) by \( k + 1 \), set \( \eta = \eta_0, m = 1 \) and go to the next step; otherwise, go to Step 8.
4. If \( f(x^k - \eta g(x^k)) - f(x^k) \leq -\frac{1}{2} \eta \|g(x^k)\|^2 \), go to Step 6; otherwise, set \( m = m + 1 \) and go to the next step.
5. Set \( \eta = \eta/2^{m-1} \) and return to Step 4.
6. Set \( x^{k+1} = x^k - \eta g(x^k) \).
7. If \( \|g(x^k)\| \leq \varepsilon \), go to Step 8; otherwise go to Step 3.
8. Output \( \{x^k; f(x^k); g(x^k)\} \).

In the case of the application of our method to imprecise problems, Algorithm 2 is replaced by other suitable algorithms as for example the corresponding one proposed in [20].

3. A perturbed dimension-reducing optimization method

The proposed method computes any critical point \( x^* \) of \( f \) (minimum, maximum or saddle) and it minimizes it if the matrix \( A_p \) of (2.6) is symmetric and positive definite so that the critical point is a minimizer. In general this matrix is not symmetric. A case where it is symmetric is given by the following lemma:

**Lemma 3.1.** If the elements \( H_{in}, i = 1, \ldots, n - 1 \), of the Hessian are equal to each other, then the matrix \( A_p \) defined by relation (2.6) is symmetric.

**Proof.** From relation (2.6) we observe that the matrix \( A_p \) is symmetric when \( \partial_n g_i = \partial_n g_j \).

In the sequel we perturb the matrix \( A_p \) using proper perturbation parameters \( \Delta_{ij}, \ i, j = 1, \ldots, n - 1 \), in order to transform it to a symmetric one. To this end we consider the mapping

\[
W = (w_1, \ldots, w_{n-1}): \mathcal{F}^* \subset \mathbb{R}^{n-1} \rightarrow \mathbb{R}^{n-1}, \text{ with } \tag{3.1}
\]

\[
w_i(y) = -\varphi_i(y) + \varphi_n(y) + \sum_{j=1}^{n-1} \Delta_{ij} x_j, \quad i = 1, \ldots, n - 1,
\]

where the matrix \( \Delta = \Delta_{ij} \) of the perturbation parameters is taken such that the inner products

\[
\langle x, \Delta_i \rangle = 0, \quad \forall x \in \mathcal{F}^* \text{ and } \forall i = 1, \ldots, n - 1, \tag{3.2}
\]

with \( \Delta_i = (\Delta_{i1}, \ldots, \Delta_{i,n-1}) \).

Now, by choosing \( \frac{1}{2} n(n-1) \) arbitrary parameters \( \Delta_{ij}, i > j \), and taking

\[
\Delta_{ij} = a_{ji} + \Delta_{ji} - a_{ij}, \quad i < j, \tag{3.3}
\]

while the remaining parameters \( \Delta_{ii} \) are computed so that \( \langle x, \Delta_i \rangle = 0 \), the matrix \( A_p = [a_{ij}] \) is transformed into a symmetric one. Since \( \Delta_{ij}, i > j \), are arbitrarily chosen, we are able to transform
$A_p$ to a diagonal one. Thus, by taking
\[ A_{ij} = -a_{ij} = \left[ -\frac{\partial_j g_i(y^p; x^{p,i}_n)}{\partial_n g_i(y^p; x^{p,i}_n)} + \frac{\partial_j g_n(y^p; x^{p,n}_n)}{\partial_n g_n(y^p; x^{p,n}_n)} \right], \quad i = 1, \ldots, n - 1, \quad i > j, \] (3.4)
from relations (3.3) and (3.2) our method becomes
\[ y^{p+1} = y^p - D_p^{-1} M_p, \quad p = 0, 1, \ldots, \] (3.5)
where $D_p$ is a diagonal matrix with elements $d_{ii}, i = 1, \ldots, n - 1$, given by
\[ d_{ii} = a_{ii} + A_{ii} = \left[ \frac{\partial_i g_i(y^p; x^{p,i}_n)}{\partial_n g_i(y^p; x^{p,i}_n)} - \frac{\partial_i g_n(y^p; x^{p,n}_n)}{\partial_n g_n(y^p; x^{p,n}_n)} \right] \\
+ \frac{1}{x_i} \sum_{j=1}^{n-1} \left\{ x_j \left[ \frac{\partial_j g_i(y^p; x^{p,j}_n)}{\partial_n g_i(y^p; x^{p,j}_n)} - \frac{\partial_j g_n(y^p; x^{p,n}_n)}{\partial_n g_n(y^p; x^{p,n}_n)} \right] \right\}. \] (3.6)
Finally, after a desired number of iterations of the above scheme, say $p = m$, the $n$th component of $x^*$ is approximated using relation (2.8).

Next, we give a proof of convergence of the perturbed dimension-reducing optimization method (3.5).

**Theorem 3.2.** Suppose that the objective function $f: \mathcal{D} \subset \mathbb{R}^n \rightarrow \mathbb{R}$ is twice-continuously differentiable in an open neighborhood $\mathcal{D}^* \subset \mathcal{D}$ of a point $x^* = (x^*_1, \ldots, x^*_n) \in \mathcal{D}$ for which $g(x^*) = \Theta^n$. Let $\mathcal{B}_i, i = 1, \ldots, n$, be those connected components of $w_i^{-1}(0)$, containing $x^*$ on which $\partial_w w_i \neq 0$ for $i = 1, \ldots, n$ respectively where the functions $w_i$ are defined by (3.1). Suppose further that
\[ \| A_{\star}^{-1} \| \| A \| < 1, \] (3.7)
where the matrix $A_{\star}$ is obtained from the matrix $A_p$ of (2.6) at $x^*$ and $A$ is the perturbation matrix. Then the iterations (3.5) and relation (2.8) will converge to $x^*$ provided the Hessian of $f$ at $x^*$ is nonsingular and the initial guess $y^0 = (x^0_1, \ldots, x^0_{n-1})$ is sufficiently close to $y^* = (x^*_1, \ldots, x^*_{n-1})$. Moreover the iterations $y^p, p = 0, 1, \ldots, \$ of (3.5) have order of convergence two.

**Proof.** Following the proof of Theorem 2.1, since the Hessian of $f$ at $x^*$ is nonsingular then $A_p$ is also nonsingular at $x^*$. Moreover, since relation (3.7) is fulfilled, then by the Banach Permutation Lemma [14] the matrix $(A_{\star} + A)$ is also nonsingular.

Using the mapping (3.1) the scheme (3.5) can be written in the following form:
\[ y^{p+1} = y^p - W(y^p)^{-1} W(y^p), \quad p = 0, 1, \ldots, \] (3.8)
where $W$ denotes the corresponding Jacobian. Clearly, we have $W = (A_{\star} + A)$ which is nonsingular at $x^*$ and, consequently, by the well-known Newton’s convergence theorem [13], for an initial guess $y^0$ sufficiently close to $y^*$, the iterations $y^p, p = 0, 1, \ldots, \$ of (3.5) converge to $y^*$ and the order of convergence is two.

Suppose now that for some $p$, for example $p = m$, we obtain $y^m = y^*$. Then, relation (2.8) yields
\[ x_n^{m+1} = \varphi_n(y^*) = x_n^*. \] (3.9)
Thus the theorem is proved. \qed
4. A finite-difference derivative dimension-reducing optimization method

In this section we consider the dimension-reducing method for unconstrained optimization using finite difference gradients and Hessian. So, suppose that the function (1.1) is continuously differentiable in the open convex domain $\mathcal{D} \subset \mathbb{R}^n$, $x \in \mathcal{D}$, and let $g(x)$ be Lipschitz continuous at $x$ in the neighborhood $\mathcal{D}$. Assume $x + he_i \in \mathcal{D}$, $i = 1, \ldots, n$, for a small quantity $h$, where $e_i$ denotes the $i$th unit vector; then using forward finite differences we obtain

$$
g_i(x) = \partial_i f \simeq \frac{f(x + he_i) - f(x)}{h}.
$$

(4.1)

Furthermore, the Hessian $H(x)$ can be approximated using only values of $f(x)$. To this end, suppose that the previous conditions regarding $f$ are satisfied and assume $x, x + he_i, x + he_j, x + he_i + he_j \in \mathcal{D}$, $1 \leq i, j \leq n$, then

$$
\partial_j g_i(x) = \partial_{ij} f \simeq \frac{f(x + he_i + he_j) - f(x + he_i) - f(x + he_j) + f(x)}{h^2}.
$$

(4.2)

For error estimates for these approximations see [4].

Using the above approximations to (2.4) and (2.8) we propose the following iterative scheme for the computation of the $n - 1$ components of an optimum $x^*$:

$$
y_{p+1} = y^p + U_p^{-1} V_p, \quad p = 0, 1, \ldots,
$$

(4.3)

where, applying (4.1) for $g_i$ instead of $f$, the elements of the matrix $U_p = [u_{ij}]$ become

$$
u_{ij} = \left[ \frac{g_i(y^p + he_j; x_n^{p,i}) - g_i(y^p; x_n^{p,i}) - g_n(y^p + he_j; x_n^{p,n}) + g_n(y^p; x_n^{p,n})}{g_i(y^p; x_n^{p,i} + he_n) - g_i(y^p; x_n^{p,i}) - g_n(y^p; x_n^{p,n} + he_n) + g_n(y^p; x_n^{p,n})} \right].
$$

(4.4)

while using (4.2) we have

$$
u_{ij} = \left[ \frac{f(y^p + he_i + he_j; x_n^{p,i}) - f(y^p + he_i; x_n^{p,i}) - f(y^p + he_j; x_n^{p,i}) + f(y^p; x_n^{p,i})}{f(y^p + he_i + he_j; x_n^{p,n} + he_n) - f(y^p + he_i; x_n^{p,n} + he_n) - f(y^p + he_j; x_n^{p,n}) + f(y^p; x_n^{p,n})} \right].
$$

(4.5)

After a desired number of iterations of (4.3), say $p = m$, the $n$th component of $x^*$ is approximated by means of one of the following relations:

$$
x_{n+1}^m = x_{n,m} - \sum_{j=1}^{n-1} \left\{ (x_j^{m+1} - x_j^m) \frac{g_n(y^m + he_j; x_n^{m,n}) - g_n(y^m; x_n^{m,n})}{g_n(y^m, x_n^{m,n} + he_n) - g_n(y^m; x_n^{m,n})} \right\},
$$

or

$$
x_{n+1} = x_{n,m} - \sum_{j=1}^{n-1} \left\{ (x_j^{m+1} - x_j^m) \right\}
$$

$$
\times \frac{f(y^m + he_j; x_n^{m,n} + he_n) - f(y^m + he_j; x_n^{m,n}) - f(y^m; x_n^{m,n} + he_n) + f(y^m; x_n^{m,n})}{f(y^m, x_n^{m,n} + 2he_n) - 2f(y^m, x_n^{m,n} + he_n) + f(y^m; x_n^{m,n})}.
$$

(4.6)

(4.7)

The signs of the gradient which are required by our method are obtained by virtue of (4.1).
5. Numerical applications

The procedures described in this paper have been implemented using a new FORTRAN program named DROPT (Dimension-Reducing OPTimization). DROPT was tested on the University of Patras HP-715 system as well as on a PC IBM compatible with random problems of various 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 indicate: \( n \) dimension, \( x^0 = (x_1, x_2, \ldots, x_n) \) starting point, \( h = (h_1, h_2, \ldots, h_n) \) starting stepsizes in each coordinate direction, \( x^* = (x_1^*, x_2^*, \ldots, x_n^*) \) approximate local optimum computed within an accuracy of \( \varepsilon = 10^{-8} \), IT the total number of iterations required to obtain \( x^* \), FE the total number of function evaluations (including derivatives), ASG the total number of algebraic signs of the components of the gradient that are required for applying the iterative schemes (2.16).

In Tables 1–3 we compare the numerical results obtained, for various starting points, by applying Armijo’s steepest descent method [1] 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. The index \( x \) indicates the classical starting point. Furthermore, D indicates divergence or nonconvergence while FR, PR and BFGS indicate the corresponding results obtained by Fletcher–Reeves [15], Polak–Ribiére [15] and Broyden–Fletcher–Goldfarb–Shanno (BFGS) [4] algorithms, respectively. Also we compare our method with well-known and efficient root-finding methods such as Brown’s method [3], a Brent–Gay modification of Brown’s method, specifically the modification of Brown’s method suggested by Brent [2] and followed by Gay (BBG) [5], as well as with Brent and Choleski–Newton methods (CN) [15]. These root-finding methods as well as our method, for the examples examined below, are applied to the following system of equations:

\[
g_i(x) = \sum_{j=1}^{n} \left( \frac{\partial f_j(x)}{\partial x_i} \right) f_j(x) = 0, \quad 1 \leq j \leq n, \tag{5.1}
\]

where \( f \) is the objective function.

Example 5.1 (Rosenbrock function [12]). This example gives the optimum for the objective function \( f \) given by

\[
f(x) = \sum_{i=1}^{2} f_i^2(x), \tag{5.2}
\]

where

\[
f_1(x) = 10 \left( x_2 - x_1^2 \right), \quad f_2(x) = 1 - x_1, \tag{5.3}
\]

with \( f(x^*) = 0 \) at \( x^* = (1, 1) \). As our starting values we utilized \( x^0 = (-1.2, 1) \) and \( h = (2, 2) \). We obtained the \( x^* = (1, 1) \) after IT = 1, FE = 4 and ASG = 20.
Table 1
Rosenbrock function

<table>
<thead>
<tr>
<th>$x^0$</th>
<th>Armijo IT</th>
<th>Armijo FE</th>
<th>FR IT</th>
<th>FR FE</th>
<th>PR IT</th>
<th>PR FE</th>
<th>BFGS IT</th>
<th>BFGS FE</th>
<th>DROPT IT</th>
<th>DROPT FE</th>
<th>ASG</th>
</tr>
</thead>
<tbody>
<tr>
<td>(−1.2, 1)$^*$</td>
<td>1881</td>
<td>21396</td>
<td>142</td>
<td>2545</td>
<td>19</td>
<td>364</td>
<td>22</td>
<td>343</td>
<td>1</td>
<td>4</td>
<td>20</td>
</tr>
<tr>
<td>(−3, 6)</td>
<td>5960</td>
<td>74560</td>
<td>194</td>
<td>4462</td>
<td>23</td>
<td>455</td>
<td>28</td>
<td>436</td>
<td>4</td>
<td>16</td>
<td>80</td>
</tr>
<tr>
<td>(−2, −2)</td>
<td>1828</td>
<td>20852</td>
<td>29</td>
<td>480</td>
<td>15</td>
<td>290</td>
<td>20</td>
<td>305</td>
<td>4</td>
<td>16</td>
<td>80</td>
</tr>
<tr>
<td>(3, 3)</td>
<td>5993</td>
<td>74364</td>
<td>130</td>
<td>2939</td>
<td>26</td>
<td>509</td>
<td>25</td>
<td>384</td>
<td>4</td>
<td>16</td>
<td>80</td>
</tr>
<tr>
<td>(1, 20)</td>
<td>D</td>
<td>D</td>
<td>259</td>
<td>5732</td>
<td>32</td>
<td>689</td>
<td>32</td>
<td>689</td>
<td>1</td>
<td>4</td>
<td>20</td>
</tr>
<tr>
<td>(10, 10)</td>
<td>18416</td>
<td>251611</td>
<td>310</td>
<td>7469</td>
<td>26</td>
<td>526</td>
<td>32</td>
<td>505</td>
<td>4</td>
<td>16</td>
<td>80</td>
</tr>
<tr>
<td>(100, 100)</td>
<td>D</td>
<td>D</td>
<td>33</td>
<td>746</td>
<td>54</td>
<td>822</td>
<td>2</td>
<td>8</td>
<td>40</td>
<td></td>
<td></td>
</tr>
<tr>
<td>(−2000, −2000)</td>
<td>2542</td>
<td>35743</td>
<td>D</td>
<td>D</td>
<td>93</td>
<td>2466</td>
<td>173</td>
<td>2667</td>
<td>2</td>
<td>8</td>
<td>40</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>$x^0$</th>
<th>Brown IT</th>
<th>Brown FE</th>
<th>BBG IT</th>
<th>BBG FE</th>
<th>Brent IT</th>
<th>Brent FE</th>
<th>CN IT</th>
<th>CN FE</th>
<th>DROPT IT</th>
<th>DROPT FE</th>
<th>ASG</th>
</tr>
</thead>
<tbody>
<tr>
<td>(−1.2, 1)$^*$</td>
<td>22</td>
<td>110</td>
<td>16</td>
<td>80</td>
<td>17</td>
<td>85</td>
<td>8</td>
<td>48</td>
<td>1</td>
<td>4</td>
<td>20</td>
</tr>
<tr>
<td>(−7, 1)</td>
<td>D</td>
<td>D</td>
<td>31</td>
<td>155</td>
<td>29</td>
<td>145</td>
<td>6</td>
<td>36</td>
<td>1</td>
<td>4</td>
<td>20</td>
</tr>
<tr>
<td>(−3, 45)</td>
<td>D</td>
<td>D</td>
<td>D</td>
<td>D</td>
<td>27</td>
<td>135</td>
<td>6</td>
<td>36</td>
<td>4</td>
<td>16</td>
<td>80</td>
</tr>
<tr>
<td>(10, 10)</td>
<td>70</td>
<td>350</td>
<td>29</td>
<td>145</td>
<td>46</td>
<td>230</td>
<td>6</td>
<td>36</td>
<td>1</td>
<td>4</td>
<td>20</td>
</tr>
<tr>
<td>(−100, 1)</td>
<td>D</td>
<td>D</td>
<td>50</td>
<td>250</td>
<td>43</td>
<td>215</td>
<td>6</td>
<td>36</td>
<td>2</td>
<td>8</td>
<td>40</td>
</tr>
<tr>
<td>(100, −100)</td>
<td>D</td>
<td>D</td>
<td>49</td>
<td>245</td>
<td>41</td>
<td>205</td>
<td>6</td>
<td>36</td>
<td>2</td>
<td>8</td>
<td>40</td>
</tr>
<tr>
<td>(100, 100)</td>
<td>D</td>
<td>D</td>
<td>D</td>
<td>D</td>
<td>D</td>
<td>6</td>
<td>36</td>
<td>2</td>
<td>8</td>
<td>40</td>
<td></td>
</tr>
</tbody>
</table>
Table 2
Freudenstein and Roth function

<table>
<thead>
<tr>
<th>$x^0$</th>
<th>Armijo</th>
<th>FR</th>
<th>PR</th>
<th>BFGS</th>
<th>DROPT</th>
<th>ASG</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>IT</td>
<td>FE</td>
<td>IT</td>
<td>FE</td>
<td>IT</td>
<td>FE</td>
</tr>
<tr>
<td>(0.5, -2)^t</td>
<td>1827</td>
<td>24155</td>
<td>18</td>
<td>356</td>
<td>8</td>
<td>187</td>
</tr>
<tr>
<td>(0.5, 1000)</td>
<td>1380</td>
<td>18770</td>
<td>D</td>
<td>D</td>
<td>D</td>
<td>D</td>
</tr>
<tr>
<td>(-2, -2)</td>
<td>1119</td>
<td>14625</td>
<td>19</td>
<td>336</td>
<td>8</td>
<td>180</td>
</tr>
<tr>
<td>(-20, 20)</td>
<td>1851</td>
<td>24986</td>
<td>24</td>
<td>451</td>
<td>10</td>
<td>211</td>
</tr>
<tr>
<td>(4.5, 4.5)</td>
<td>1239</td>
<td>16289</td>
<td>18</td>
<td>342</td>
<td>9</td>
<td>196</td>
</tr>
<tr>
<td>(10, 100)</td>
<td>1845</td>
<td>24664</td>
<td>10</td>
<td>200</td>
<td>9</td>
<td>194</td>
</tr>
<tr>
<td>(12, 2)</td>
<td>2027</td>
<td>26886</td>
<td>70</td>
<td>1145</td>
<td>8</td>
<td>130</td>
</tr>
<tr>
<td>(4, -1000)</td>
<td>1886</td>
<td>25597</td>
<td>D</td>
<td>D</td>
<td>D</td>
<td>D</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>$x^0$</th>
<th>Brown</th>
<th>BBG</th>
<th>Brent</th>
<th>CN</th>
<th>DROPT</th>
<th>ASG</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>IT</td>
<td>FE</td>
<td>IT</td>
<td>FE</td>
<td>IT</td>
<td>FE</td>
</tr>
<tr>
<td>(0.5, -2)^t</td>
<td>5</td>
<td>25</td>
<td>6</td>
<td>30</td>
<td>6</td>
<td>30</td>
</tr>
<tr>
<td>(0.5, 1000)</td>
<td>D</td>
<td>D</td>
<td>D</td>
<td>D</td>
<td>D</td>
<td>D</td>
</tr>
<tr>
<td>(-20, -200)</td>
<td>D</td>
<td>D</td>
<td>D</td>
<td>D</td>
<td>D</td>
<td>D</td>
</tr>
<tr>
<td>(4.5, -8)</td>
<td>D</td>
<td>D</td>
<td>D</td>
<td>15</td>
<td>75</td>
<td>14</td>
</tr>
<tr>
<td>(10, -20)</td>
<td>D</td>
<td>D</td>
<td>21</td>
<td>105</td>
<td>D</td>
<td>D</td>
</tr>
<tr>
<td>(12, -24)</td>
<td>38</td>
<td>190</td>
<td>D</td>
<td>D</td>
<td>D</td>
<td>D</td>
</tr>
<tr>
<td>(4, -1000)</td>
<td>D</td>
<td>D</td>
<td>D</td>
<td>D</td>
<td>D</td>
<td>D</td>
</tr>
<tr>
<td>(10,100)</td>
<td>D</td>
<td>D</td>
<td>D</td>
<td>D</td>
<td>D</td>
<td>D</td>
</tr>
</tbody>
</table>

Example 5.2 (Freudenstein and Roth function [12]). In this example the objective function $f$ is given by

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

(5.4)

where

$$f_1(x) = -13 + x_1 + ((5 - x_2)x_2 - 2)x_2, \quad f_2(x) = -29 + x_1 + ((x_2 + 1)x_2 - 14)x_2,$$

(5.5)

with $f(x^*) = 0$ at $x^* = (5, 4)$ and $f(x^*) = 48.9842 \ldots$ at $x^* = (11.41 \ldots, -0.8968 \ldots)$. As our starting values we utilized $x^0 = (0.5, -2)$ and $h = (2, 2)$. We obtained the $x^* = (5, 4)$ after IT = 3, FE = 12 and ASG = 60.

Example 5.3 (Brown almost-linear function [12]). In this case the objective function $f$ is given by

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

(5.6)
### Table 3
Brown almost-linear function

<table>
<thead>
<tr>
<th>$x^0$</th>
<th>Armijo</th>
<th></th>
<th>FR</th>
<th></th>
<th>PR</th>
<th></th>
<th>BFGS</th>
<th></th>
<th>DROPT</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>IT</td>
<td>FE</td>
<td>IT</td>
<td>FE</td>
<td>IT</td>
<td>FE</td>
<td>IT</td>
<td>FE</td>
<td>IT</td>
<td>FE</td>
</tr>
<tr>
<td>$(0.5, 0.5, 0.5)^2$</td>
<td>177</td>
<td>1330</td>
<td>12</td>
<td>177</td>
<td>6</td>
<td>93</td>
<td>6</td>
<td>91</td>
<td>6</td>
<td>54</td>
</tr>
<tr>
<td>$(0, 0, 3)$</td>
<td>221</td>
<td>1612</td>
<td>27</td>
<td>389</td>
<td>9</td>
<td>143</td>
<td>9</td>
<td>131</td>
<td>1</td>
<td>9</td>
</tr>
<tr>
<td>$( -1, 0, 3)$</td>
<td>233</td>
<td>1691</td>
<td>34</td>
<td>491</td>
<td>42</td>
<td>612</td>
<td>12</td>
<td>180</td>
<td>1</td>
<td>9</td>
</tr>
<tr>
<td>$(0.1, 0.1, -2)$</td>
<td>181</td>
<td>1349</td>
<td>52</td>
<td>742</td>
<td>9</td>
<td>149</td>
<td>9</td>
<td>145</td>
<td>4</td>
<td>36</td>
</tr>
<tr>
<td>$(1.2, 1.2, 0)$</td>
<td>183</td>
<td>1368</td>
<td>23</td>
<td>334</td>
<td>9</td>
<td>138</td>
<td>8</td>
<td>121</td>
<td>3</td>
<td>27</td>
</tr>
<tr>
<td>$(0.8, 0.7, -2)$</td>
<td>193</td>
<td>1446</td>
<td>70</td>
<td>996</td>
<td>23</td>
<td>336</td>
<td>14</td>
<td>216</td>
<td>4</td>
<td>36</td>
</tr>
<tr>
<td>$( -0.1, -0.1, -0.1)$</td>
<td>173</td>
<td>1295</td>
<td>20</td>
<td>290</td>
<td>7</td>
<td>109</td>
<td>6</td>
<td>92</td>
<td>3</td>
<td>27</td>
</tr>
<tr>
<td>$(1.2, 1.2, 10)$</td>
<td>64</td>
<td>563</td>
<td>47</td>
<td>711</td>
<td>10</td>
<td>165</td>
<td>9</td>
<td>139</td>
<td>3</td>
<td>27</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>$x_0$</th>
<th>Brown</th>
<th></th>
<th>BBG</th>
<th></th>
<th>Brent</th>
<th></th>
<th>CN</th>
<th></th>
<th>DROPT</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>IT</td>
<td>FE</td>
<td>IT</td>
<td>FE</td>
<td>IT</td>
<td>FE</td>
<td>IT</td>
<td>FE</td>
<td>IT</td>
<td>FE</td>
</tr>
<tr>
<td>$(0.5, 0.5, 0.5)^2$</td>
<td>7</td>
<td>63</td>
<td>8</td>
<td>72</td>
<td>8</td>
<td>72</td>
<td>9</td>
<td>108</td>
<td>6</td>
<td>54</td>
</tr>
<tr>
<td>$(0, 0, 17)$</td>
<td>13</td>
<td>117</td>
<td>22</td>
<td>198</td>
<td>34</td>
<td>306</td>
<td>D</td>
<td>D</td>
<td>1</td>
<td>9</td>
</tr>
<tr>
<td>$( -1, 0, -1)$</td>
<td>10</td>
<td>90</td>
<td>17</td>
<td>153</td>
<td>D</td>
<td>D</td>
<td>9</td>
<td>108</td>
<td>1</td>
<td>9</td>
</tr>
<tr>
<td>$( -1, 0, 23)$</td>
<td>11</td>
<td>99</td>
<td>33</td>
<td>297</td>
<td>22</td>
<td>198</td>
<td>D</td>
<td>D</td>
<td>1</td>
<td>9</td>
</tr>
<tr>
<td>$(0.8, 0.7, -0.7)$</td>
<td>30</td>
<td>270</td>
<td>D</td>
<td>D</td>
<td>D</td>
<td>D</td>
<td>D</td>
<td>D</td>
<td>4</td>
<td>36</td>
</tr>
<tr>
<td>$(0.8, 0.7, -1.7)$</td>
<td>7</td>
<td>63</td>
<td>33</td>
<td>297</td>
<td>13</td>
<td>117</td>
<td>15</td>
<td>180</td>
<td>4</td>
<td>36</td>
</tr>
<tr>
<td>$(1.2, 1.2, -10)$</td>
<td>50</td>
<td>450</td>
<td>19</td>
<td>171</td>
<td>D</td>
<td>D</td>
<td>D</td>
<td>D</td>
<td>3</td>
<td>27</td>
</tr>
<tr>
<td>$(1.2, 1.2, -23)$</td>
<td>14</td>
<td>126</td>
<td>30</td>
<td>270</td>
<td>23</td>
<td>207</td>
<td>D</td>
<td>D</td>
<td>3</td>
<td>27</td>
</tr>
</tbody>
</table>
Table 4
Rosenbrock function

<table>
<thead>
<tr>
<th>$x^0$</th>
<th>Brown</th>
<th></th>
<th>BFGS</th>
<th></th>
<th>DROPT</th>
<th></th>
<th>FDDROPT</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>IT</td>
<td>FE</td>
<td>IT</td>
<td>FE</td>
<td>IT</td>
<td>FE</td>
<td>ASG</td>
</tr>
<tr>
<td>$(−1.2, 1)^2$</td>
<td>22</td>
<td>210</td>
<td>22</td>
<td>343</td>
<td>1</td>
<td>4</td>
<td>20</td>
</tr>
<tr>
<td>$(−3, 6)$</td>
<td>D</td>
<td>D</td>
<td>28</td>
<td>436</td>
<td>4</td>
<td>16</td>
<td>80</td>
</tr>
<tr>
<td>$(−3, 45)$</td>
<td>D</td>
<td>D</td>
<td>51</td>
<td>771</td>
<td>4</td>
<td>16</td>
<td>80</td>
</tr>
<tr>
<td>$(100, 1)$</td>
<td>D</td>
<td>D</td>
<td>26</td>
<td>410</td>
<td>1</td>
<td>4</td>
<td>20</td>
</tr>
<tr>
<td>$(0.7, −4)$</td>
<td>15</td>
<td>75</td>
<td>45</td>
<td>682</td>
<td>4</td>
<td>16</td>
<td>80</td>
</tr>
<tr>
<td>$(0.5, −5)$</td>
<td>16</td>
<td>80</td>
<td>51</td>
<td>783</td>
<td>6</td>
<td>24</td>
<td>120</td>
</tr>
<tr>
<td>$(0.8, 3)$</td>
<td>D</td>
<td>D</td>
<td>16</td>
<td>235</td>
<td>4</td>
<td>16</td>
<td>80</td>
</tr>
<tr>
<td>$(1, 2)$</td>
<td>17</td>
<td>85</td>
<td>13</td>
<td>193</td>
<td>1</td>
<td>4</td>
<td>20</td>
</tr>
</tbody>
</table>

where

$$f_i(x) = x_i + \sum_{j=1}^{n} x_j - (n + 1), \quad 1 \leq i < n, \quad f_n(x) = \left( \prod_{j=1}^{n} x_j \right) - 1,$$

(5.7)

with $f(x^*) = 0$ at $x^* = (a, \ldots, a, a^{1−n})$ where $a$ satisfies the equation $na^n - (n + 1)a^{n−1} + 1 = 0$ and $f(x^*) = 1$ at $x^* = (0, \ldots, 0, n + 1)$. As our starting values we utilized $n = 3$, $x^0 = (0.5, 0.5, 0.5)$ and $h = (2, 2, 2)$. We obtained the $x^* = (0.76759187, 0.76759187, 1.69722437)$ after IT = 6, FE = 54 and ASG = 180.

Next, we apply our method using finite differences (FDDROPT) and we compare these results with the corresponding ones obtained by Brown, DROPT and BFGS methods without finite differences for various starting guesses. These results are shown in Table 4.

6. Concluding remarks

This paper describes a new efficient numerical method for computing an unconstrained local optimum. This method rapidly minimizes general functions and it appears to be superior to other well-known optimization and root-finding methods on a variety of classical test functions. We have tested our method for higher dimensions and we have observed relative results. In the case of large sparse problems, if, for some reason, the component $x_{int}$ which corresponds to the dimension-reducing component is missing for some component $g_i$, we replace it by the function $g_i + g_k$ where $g_k$ includes $x_{int}$.

Although the method of this paper uses reduction to simpler one-dimensional equations, it converges quadratically to $n − 1$ components of the optimum, while the remaining component of the optimum is evaluated separately using the final approximations of the other components. Thus, it does not require a good initial estimate for one component of the optimum. Besides, this method does not directly perform gradient evaluations, since it uses the modified one-dimensional bisection
method. It requires only that the algebraic signs of the function and gradient values be correct, so that it can be applied to problems with imprecise function values.

Since in general the matrix of our reduced system is not symmetric, we have transformed it to a symmetric one by using proper perturbations. Also, applying this transformation we have been able to obtain analytical forms of the equivalent diagonal matrix.

Furthermore, we have substituted finite difference approximations for the elements of the matrix $A_R$ and we give the corresponding dimension-reducing scheme. By means of this, we have been able to compute optimum points utilizing only values of the objective function.

Acknowledgements

We wish to acknowledge the referee for useful suggestions.

References


