A VARIABLE ORDER ALGORITHM FOR UNCONSTRAINED MINIMIZATION

by

A-J. Jimenez & S.W. Director

DRC-1S-3-79

January 1979

* IBM
System Products Division
East Fishkill, NY 12533

** Department of Electrical Engineering
Carnegie-Melion University
Pittsburgh, PA 15213

This work was supported in part by NSF Grant No. ENG-75-23078,

ABSTRACT

A variable order algorithm is proposed for *the* rini^ization of a function cff several variables. This algorithm has an order of convergence *.s high as four, good accuracy, and features a scalar subproblem at each iteration which may be along curved trajectories in the space *of* the independent variables. Numerical results en five standard test problems indicate superior performance when compared with seven other popular algorithms.

I. Introduction

A new iterative algorithm, called the Variable Order (VO) algorithm, has been developed for finding a local minimum solution to the problem

$$\underset{x}{\text{minimize}}\ f(x), \tag{1}$$

where $f(x)$ is a reasonably smooth scalar function of n independent variables, $f:E^n \rightarrow E$. The VO algorithm has to the authors' knowledge several properties and features not found in any previously published algorithm. Among the new features are that the order of convergence may be as high as four, and that the scalar search at each iteration may be along curved trajectories in the space of the independent variables. Values of the function, and its first two derivatives (i.e., the gradient and the hessian) are required by the algorithm, however these quantities may be approximated using differencing techniques. Numerical results have shown that, in spite of the effort required to compute the hessian, for a large class of functions the VO algorithm is considerably more efficient than many existing algorithms which do not require this computation.

It will be convenient to describe the VO algorithm in terms of the general iteration given by

$$x^{k+1} = H^k(p_k, x^k), \quad k = 0, 1, 2, \ldots, \tag{2}$$

where H^k is the k^{th} iteration function. This iteration begins from an initial guess $\tilde{x}^0$ and a sequence $\{\tilde{x}^k\}$ is generated which hopefully converges to a solution $\tilde{x}^*$ of (1). Two principal steps are implicit in (2): the transformation step and the scalar search step. The transformation step consists of computing the transformation function given by

$$h^k(p) = JJ^k(p, x^k) \ . \tag{3}$$

The transformation function represents a direction or a trajectory along which the next point $\underset{\sim}{x}^{k+1}$ is selected. The transformation step for the VO algorithm is described in more detail in the next section. The scalar search step consists of selecting the next point given by

$$\underset{\sim}{x}^{k+1} = \underset{\sim}{h}^{k}(p_k), \tag{4}$$

where the scalar parameter value $p = p_k$ is suitably selected. The scalar search step is described in more detail in Section 3. An example is given in Section 4 which illustrates the generally curved trajectories along which the scalar search is undertaken. Section 5 offers an outline of the details of the algorithm. The paper ends with a comparison of the VO algorithm with seven other popular minimization algorithms when they are all used to minimize five standard test functions.

2. The Transformation Step

It is well-known that a necessary condition for a local minimum solution of (1) is that the gradient of $f(\underset{\sim}{x})$ must be zero [1]. In this section we develop a family of transformation functions, $\underset{\sim}{h}^{k}(p)$, with an increasing order of convergence by considering the truncation of the infinite series expansion of the point $\underset{\sim}{x}^*$ at which the gradient is zero about the k^{th} estimate of the solution $\underset{\sim}{x}^{k}$.

Because it is a necessary condition, we are interested in solutions to

$$\underset{\sim}{f}'(\underset{\sim}{x}) = \underset{\sim}{0}\ , \tag{5}$$

where $\underset{\sim}{f}'(\underset{\sim}{x})$ is the column vector of first partial derivatives of $f(\underset{\sim}{x})$ (i.e., the transpose of the gradient). Consider the change of variables denoted by

$$\underset{\sim}{x} = \underset{\sim}{X}(\underset{\sim}{z})\ , \tag{6}$$

where $\underset{\sim}{X}$ may be a nonlinear vector function. Using (6), we obtain

$$\underset{\sim}{f}'(\underset{\sim}{x}) = \underset{\sim}{f}'(\underset{\sim}{X}(\underset{\sim}{z})) = \underset{\sim}{g}(\underset{\sim}{z}) \ . \qquad (7)$$

Let $\underset{\sim}{x}^*$ satisfy (5) and define $\underset{\sim}{z}^*$ such that (6) is satisfied. Then from (7)

$$g(z^*\} = 0 \ . \qquad (8)$$

If the above system of equations is simple to solve then ^* mav be readily obtained,

and the point $\underset{\sim}{x}^*$ may be computed from (6) by letting $\underset{\sim}{z}=\underset{\sim}{z}^*$. Clearly specifying a change of variables which yields a function $\underset{\sim}{g}$ for which (8) is simple to solve could be difficult. However we can start by specifying a suitable function $\underset{\sim}{c}j$ and determine the resulting function $\underset{\sim}{X}$ from a Taylor series expansion. One suitable class of functions is[1]

#>•<*?• 4. . . . ZS'T • <>

where p is a scalar parameter greater than zero.[2] Mote that for this function, (8) is simple to solve; in particular $\underset{\sim}{z}^*=\underset{\sim}{0}$. Consider the first two terms of the Taylor series expansion of (6) given by

$$\underset{\sim}{X} \doteq \underset{\sim}{X}(\underset{\sim}{z}^k) + \underset{\sim}{X}'(\underset{\sim}{z}^k)\ (\underset{\sim}{z} - \underset{\sim}{z}^k) \ , \qquad (10)$$

where a z^k may be associated with $\$'$ throuqh (7);

---

[1] Other functions can also bo used which yield different series exnansinns : one of these expansions is the n-dimpnsional extension o^ a nreviousl*' kno^n scries attributed to Euler, Jimenez (1976), obtained with p=l in (9).

[2] z_1° - denotes z_1 - raised to the n^{th} nov/er.

and

$$X'(z^k) = f''(x^k) \cdot g'(z^k) \qquad \text{(ID}$$

is obtained by differentiating (7) with respect to £, and where $f''(x^k)^{-1}$ is the inverse of the hessian matrix of f evaluated at $x=£^k$. If we use the g function (9), reconnize that $J j' \wedge V = \eta(z^k) = \wedge'(x^k)$ and that (10) 1-s an approximation, at %*%*=%, one obtains the iterative method

$$x^{k+1} = x^k - p_k\, f''(x^k)^{-1}\, f'(x^k) \qquad (12)$$

for some suitable value $p=p_k$- This iteration is Newton's algorithm for solving (1) [1] and is characterized by the second-order transformation function given by

$$\text{.»»» •• 4}^{k} - \text{p «2 •} \qquad (13)$$

where the second-order correction, d_2^k, is defined as the solution of the system of linear equations given by

$$f''(x^k)\, d_2^k = f'(x^k) \,. \qquad (14)$$

If three terms of the Taylor series expansion of (6), using (9), are kept, we obtain the third-order transformation given by

$$h_3^k(p) = x^k - \frac{3}{2} d_2^k\, p - (d_3^k - \frac{1}{2} d_2^k)\, p^2 \,, \qquad (15)$$

where the third-order correction, d^M, is the solution of the linear system of equations given by

$$f;(x^k)\, d] = (1/2)\, f'''(x^k)\, djdj \,,$$

and $f'''(x^k)$ is the third derivative tensor of f evaluated at $x=x^k$. Fortunately

this term need not be evaluated exactly. It can be shown that by considering a Taylor series expansion of $f'(x^k - d_0)$ about $x=x^K$, the third-order correction may be approximated by solving the system of equations

$$\underset{\sim}{f}''(\underset{\sim}{x}^k)\underset{\sim}{d}_3^k \doteq \underset{\sim}{f}'(\underset{\sim}{h}_2^k(1)) = \underset{\sim}{f}'(\underset{\sim}{x}^k - \underset{\sim}{d}_2^k) \ . \tag{16}$$

If four terms of the Taylor series expansion of (6), using (9), are retained, we obtain the fourth-order transformation given by

$$\cdot,4^{(p)} = Z \quad - Tfe^{P} \cdot \{\wedge 3 - y\ P \quad - llU - h^{+}\ 6^{-}\wedge 2)^{P} \quad , \tag{17}$$

where the fourth-order correction, $\underset{\sim}{d}_4$, is the solution of the linear system of equations given by

$$f\ (x^{\flat}\ ,\ d,\ = f\ ,\ x\ ,\ d_0\ d\ll\ - r\ f\ \ (x\ )d_0\ d_0\ d_0\ ,$$

and f (x) is the fourth derivative tensor of f evaluated at $x=x^k$. As before the fourth-order correction need not be evaluated exactly, but by considering a Taylor series expansion of $/'(x^k - d_0 - cU)$ about $x=x^k$ can be approximated by solving

$$\underset{\sim}{f}''(\underset{\sim}{x}^k)\underset{\sim}{d}_4^k \doteq \underset{\sim}{f}'(\underset{\sim}{h}_3^k(1)) = \underset{\sim}{f}'(\underset{\sim}{x}^k - \underset{\sim}{d}_2^k - \underset{\sim}{d}_3^k) \ . \tag{18}$$

Transformation functions of order higher than four may be similarly derived. However, it does not seem possible to adequately approximate the corrections of order higher than four, -and since higher-order derivatives are computationally expensive to evaluate and require considerable storage they should be avoided.[3]

Before leaving this section there are three points which need to be discussed: 1) approximation of the hessian and the gradient, 2) singularity

of the hessian matrix, and 3) selection of the transformation order at each iteration.

Evaluation of the corrections as dictated by expressions (14), (16) and (18) requires that both the gradient and the hessian be evaluated. If explicit expressions were in fact needed, the usefulness of,the algorithm would be severely limited. However, good results have been obtained when one or both of these derivatives are approximated using differencing techniques. Unfortunately, the use of a quasi-Newton method of approximating the hessian inverse [1] is apparently not sufficiently accurate for the Drooosed ainorithm as experimental evidence has shown.

Given the gradient and the hessian, or their approximations, the high-order corrections are computed by solving the linear systems of equations (14), (16), and (13). Observe that these systems of equations have the same coefficient matrix: the hessian evaluated at $\underset{\sim}{x}$. Thus it is efficient to compute the Cholesky factorization [3] of the hessian just once which allows the computation of all the high-order corrections by the procedure of forward and back substitution. In computing the Cholesky factorization, one must consider the possibilities that the hessian may be singular or may be negative definite at $\underset{\sim}{x}^k$. If tha hessian is singular, the factorization of the hessian does not exist. If the hessian is negative definite, the infinite series expansion of $\underset{\sim}{x}^*$, the point at which the gradient is zero and the point used to derive the transformation functions, is likely to be a local maximum or a saddle point, since the gradient is also zero at these points. Thus the Cholesky factorization to be used modifies the hessian, whenever it is not positive definite, by in effect solving $\underset{\sim}{d}$« from

$$[\underset{\sim}{f}''(\underset{\sim}{x}^k) + \underset{\sim}{D}^k]\underset{\sim}{d}\hat{} = \underset{\sim}{f}'(\underset{\sim}{x}^k) \ , \qquad (19)$$

where $D^{\wedge}$ is a diaoonal matrix. This nodified factorization is a variation of the one nronosed by Murray (1972) by adding a diagonal oivoting strategy to the procedure which introduces fewer nonzero diaaonal elements to Q* than Murray's orocedure (see Jimenez (1976)).

The transformation order selected at the k^{th} iteration is based on the convenience of the series expansion of (6), for o=l in (9), and uoon obtaining functional descent. That is, if the point obtained from the transformation function for p=l yields a function value less than the current value of the function, then the next transformation order is considered. Figure 1 summarizes this selection procedure in a flowchart.

3. The Scalar Search Step

Given a particular transformation function, the scalar search step involves determination of a value of the scalar parameter p appearing in the transformation function such that

$$f(\underset{\sim}{h}^{\wedge}(p)) < f(\underset{\sim}{x}^{k}) = f(\underset{\sim}{h}J;(0)), \quad r = 2, 3, \text{ or } 4. \tag{20}$$

In many reported minimization algorithms this step is *very* time consuming. The source of the difficulty is the requirement that a value of p be found which accurately solves the scalar minimization problem given by [1]

$$\underset{P}{\text{minimize}} \quad f(h^{k}(p)) \quad . \tag{21}$$

In fact, several studies have indicated that the overall efficiency of many minimization algorithms is quite sensitive to hovr accurate the solution of (21) is comnuted (e.g., [4] and [1]). In addition,

it has been the authors[1] experience[4] that choosing a p in this manner tends to force most algorithms to follow the bottom of narrow valleys, i* present, with relatively slow orogress towards the solution.

Since the VO algorithm docs not require an accurate solution of (21) a new strategy is employed. This strategy is based upon determining the "closeness[11] of the current estimate, $\underset{\sim}{x}^k$, to the solution, $\underset{\sim}{x}^*$. We define $\underset{\sim}{x}^k$ as being "close" to %* if the function has a quadratic behavior at $£^k$, which is defined by

$$||\underset{\sim}{f}'(\underset{\sim r}{h}^k(1))|| \leq \varepsilon_c \ . \qquad (22)$$

(For the functions tested, which are not badly scaled, $e_c{}^s l$ was reasonable.) If the current point $\underset{\sim}{x}^k$ is close to a local minimum, the p_k is computed to solve (21). n_n the other hand if $£^k$ is <u>not</u> close to a local minimum, then p_k is computed under the heuristic principle that $\underset{\sim}{x}^{k+1} = lj\wedge(P_k)$ be as far from$\wedge^k$ as possible, as long as $f(\underset{\sim}{x}^{k+1})$ is less than $f(\wedge)$. Each of these possibilities will be described in more detail next.

If x^k is close to x*, the solution of (21) is approximated in a standard manner by bracketing the minimum along the trajectory represented by the transfonnation function and computing the minimum of the quadratic in p through three function values. As it might be expected from the relationship of the proposed algorithm with Newton's method, the solution of (21) is p=l for most local minima; thus solving (21) is not time consuming when x is close to x*.

When (22) is not satisfied, the proposed heuristic principle may be mathematically described by

$$\underset{p}{\text{maximize}} \quad \| \hat{}(p) - \underset{\sim}{x}^k \| \quad , \tag{23a}$$

$$\text{subject to} \quad f(\hat{}(p)) < f(\underset{\sim}{x}^k) - C_k \quad , \tag{23b}$$

where $C_k \geq 0$ is defined to insure that $f(\underset{\sim}{x}^{k+1})$ is sufficiently less than $f(x^k)$.

The technique for approximating a solution to (23) depends on which transformation order was selected, i.e., the value of r. If the second-order transformation was selected, (23a) is linear in p and therefore (23a) is generally maximized by the largest value of p which satisfies (23b). The procedure consists of fitting and computing the minimum of approximating polynomials which attempts to satisfy (23b). Then attempting to satisfy (23a), a constant is added to the computed minimum of the approximating polynomial. The function is then evaluated at the resulting value of p, and if (23b) is satisfied, the search is complete, otherwise the nrocedure is repeated. This technique is summarized in Figure 2.

If the third- or the fourth-order transformation is selected, (23a) is no longer linear in p. Therefore, the solution of (23) is along a curved trajectory in the space of the independent variables as the illustrative example in the next section shows. For clarity of notation we will drop the order subscript, r, and the iteration superscript, k, from the transformation function for the remainder of this section. Since x^{k+1} Mp«,)> the th coordinate of all the possible points that may become x^{k+1} is given by $h_i(p)$. Since $h_i(p)$ is not linear in p, the i^{th} coordinate initially roves away from the current value as p is increased from p=0, and it may then approach the current value after p exceeds some magnitude. That is, the quantity

$$(x_i^{k+1} - x_i^k)^2 = (h_i(p) - x_i^k)^2$$

may have stationary points which must satisfy

$$h_i'(p) = 0, \tag{24}$$

where $h_i'(p)$ is the derivative with respect to p. Equation (24) is linear in p for the third-order transformation, and quadratic in p for the fourth-order transformation. Therefore, these significant values of p may be easily found, and those which are positive are candidates to satisfy (23). It is proposed that these significant values of p be computed for all coordinates by the use of (24), discardinn any which are not positive. Moreover, it was experimentally found for the tested functions that p_k is generally not greater than six for the V0 algorithm. The function is then evaluated at each of the significant values beginning with the largest and proceeding to the smallest and as soon as the descent contraint (23b) is satisfied, the scalar search is complete. In case none of these values of p *are* positive and smaller than six, the function is evaluated for increasing values of p until (23b) is satisfied for the largest value of p. For all the functions tested, in most iterations (24) yielded values within the acceptable range. Furhtermore, in most iterations only one additional function evaluation was needed to end the scalar search.

4. IIlustrative Example

The problem proposed by Rosenbrock [1] is chosen to illustrate the V0 algorithm features. This problem is given by

$$\underset{\underset{\sim}{x}}{\text{minimize}} \quad f(x_1, x_2) = 100(x_2 - x_1^2)^2 + (1 - x_1)^2 . \tag{25}$$

The usual starting point, $\underset{\sim}{x}^0 = (-1.2,\ 1)^T$, will be used. At this point, the second-order transformation is given by

$$\underset{\sim}{h}_2^0(p) = \begin{bmatrix} -1.2 \\ \\ 1 \end{bmatrix} - p \begin{bmatrix} -.024719 \\ \\ -.3807 \end{bmatrix} \qquad (26)$$

The third-order transformation is given by

$$\underset{\sim}{h}_3^0(p) = \begin{bmatrix} -1.2 \\ \\ 1 \end{bmatrix} - p \begin{bmatrix} -.03708 \\ \\ .571 \end{bmatrix} - p^2 \begin{bmatrix} -.01205 \\ \\ .2483 \end{bmatrix}. \qquad (27)$$

And finally, the fourth-order transformation is given by

$$\underset{\sim}{h}_4^0(p) = \begin{bmatrix} -1.2 \\ \\ 1 \end{bmatrix} - p \begin{bmatrix} -.0453 \\ \\ -.6979 \end{bmatrix} - p^2 \begin{bmatrix} -.0241 \\ \\ .4966 \end{bmatrix} - p^3 \begin{bmatrix} -.00368 \\ \\ -.0657 \end{bmatrix}. \qquad (28)$$

Figure 3 illustrates the x_2, x_1 plane with equi-contours of Rosenbrock's function shown as dashed lines. Each of the three curves emanating from $\underset{\sim}{x}^0$ corresponds to a trajectory generated by the three transformations (26), (27), and (28) as p is increased. Clearly, the fourth-order transformation is the best, and it is the one that the proposed selection procedure outlined in Fig. 1 chooses. The scalar search, outlined in the preceding section, computes $p_k = 4.1957$ requiring only one additional function evaluation to yield the next point to be

$$\underset{\sim}{x}^1 = \underset{\sim}{h}_4^0(4.1957) = (-.3138, .03796)^T .$$

5. Algorithm

The VO algorithm can now be summarized. (For a detailed listing see [2]).

STEP 1: INITIALIZATION. Set k=0 and obtain a value for $\underset{\sim}{x}^0$. Evaluate $f(\underset{\sim}{x}^0)$ and $\underset{\sim}{f}'(\underset{\sim}{x}^0)$ or its approximation.

STEP 2: TRANSFORMATION. This step may be subdivided as follows:

a. Compute the hessian $\underset{\sim}{f}''(\underset{\sim}{x}^k)$ or its approximation.

b. Compute the modified Cholesky factors of the hessian.

c. Compute the second-order correction, c^k, given by (14).

d. Compute $f(x^k - d_2^k)$ and $f'(x^k - d_2^k)$.

e. if $\|f'(x^k - d_2^k)\| \leq 10^{-4}$, set $x^* = x^k - d_2^k$ and exit (we are done).

f. If $f(x^k - d_2^k) < f(x^k)$, set order r=2, and go to STEP 3, otherwise continue.

g. Compute the approximation to the third-order correction, d_3^k, given by (16).

h. Compute $f(x^k - d_2^k - d_3^k)$ and $f'(x^k - d_2^k - d_3^k)$.

i. If $\|f'(x^k - d_2^k - d_3^k)\| \leq 10^{-4}$, set $x^* = x^k - d_2^k - d_3^k$ and exit.

j. If $f(x^k - d_2^k - d_3^k) < f(x^k)$, set order r=2, and go to STEP 3, otherwise continue,

k. Compute the approximation to the fourth-order correction, d_4^k, given by (13).

l. Compute $f(x^k - d_2^k - d_3^k - d_4^k)$.

m. If $f(x^k - d_2^k - d_3^k - d_4^k) > f(x^k)$, set order r=3, else set order r=4.

STEP 3: SCALAR SEARCH. If the order r is 2, execute a, else execute b.

a. If $f(x^k - d_2^k) < f(x^k)$, set $x^{k+1} = x^k - d_2^k$, and go to STEP 4. Otherwise approximate the solution of (23) as described in Section 3 to obtain p_k. Set $x^{k+1} = h_2^k(P_k)$ and go to STEP 4.

b. If $|f'(h(1))| < 1$, then compute the solution of the scalar minimization problem (21), otherwise approximate the solution of (23) to obtain p_R. Set $x^{k+1} = J_r^k(P_k)$ and go to STEP 4.

STEP 4: CONVERGENCE TEST. Compute $f'(x^{k+1})$ if it is not already available. If $\|f'(x^{k+1})\| < \text{STEPS}$, and the hessian is positive definite, set $x^* = x^{k+1}$, and exit. Otherwise, set k=k+1, and GO to STEP 2.

6. Numerical Results

Five widely used test problems were selected to test and compare the VO algorithm with other popular algorithms. We briefly illustrated the algorithm with Rosenbrock's problem in Section 4. In Table 1 we give the entire history of the iterations required to compute the minimum of this problem. Mote that even with only function values supplied, the algorithm is able to obtain the minimum *very* accurately. Figure 4 shows the trajectory from the initial point to the neighborhood of the minimum.

The other test problems are the following. Powell's singular hessian at the solution, [12], given by

$$f(\underset{\sim}{x}) = (x\text{^}l0xg)^2 + 5(x_3-x_4)^2 + (x_2-2x_3)^4 + 10(x_r x_4)^4 \ , \qquad (29)$$

with initial point $\underset{\sim}{x}^\circ=(3, -1, 0, 1)^T$. Fletcher and Powell's (1963) helical valley problem given by

$$f(\underset{\sim}{x}) - 100[(x_3-10\ 6)^2 + (R - l)^2] + x_3^2 \qquad (30)$$

where

$$-TT/2 < 2TTO = tarf\text{^}Xg/x_{\text{1}}) < 3TT/2 \ ,$$

$$R = (x_1^2 + x_2^2) \ ,$$

with initial point $\underset{\sim}{x}^0=(-1, 0, 0)^T$. Wood's saddle point problem, [13], given by

$$f(\underset{\sim}{x}) = 100(x_2-x_1^2)^2 + (1-x_l)^2 + 90(x_4-x_3^2)^2 + (l-x_3)^2$$
$$+ 10.1[(x_2-l)^2 + (x_4-!)^2] + 19.8(x_2-l)(x_4-l) \ , \qquad (31)$$

with initial point $\text{^}^0=(-3, -1, -3, -1)^T$. Cragg and Levy's singular hessian at the solution [14] given by

$$f(x) = (e^{x_1} - x_2)^4 + 100(x_2 - x_3)^6 + \tan^4(x_3 - x_4) + x_1^8 + (x_4 - 1)^2, \quad (32)$$

with initial point $x^0 = (1, 2, 2, 2)^T$.

Table 2 summarizes the terminating counters and the convergence data for the VO algorithm for the five test problems. Published results can be used to compare seven popular algorithms with the VO algorithm. The existing algorithms which are used in the comparisons include the following:

FR - Fletcher and Reeves conjugate gradients algorithm [15].

DFP - Davidon--Fletcher and Powell quasi-Newton with a rank 2 update [16] and [17]..

B - Broyden quasi-Newton with a rank 1 update [18].

F - Fletcher quasi-Newton with a combination of rank 1 and rank 2 updates [19].

P - Powell conjugate directions with functions only [12].

S - Stewart DFP with function values only [20].

C - Cullum B with function values only [21] and [22].

Almost all the results for the above algorithms an the five test problems are obtained from a comparative study published by Himmelblau [5] and [6]. The exceptions are the following. The C -algorithm was not compared by Himmelblau, and thus Cullum's results are used for the three problems reported by her; the results for Rosenbrock's problem were not tabulated by Himmelblau, and therefore the original publication results are used, or the results published by Sargent and Sebastian [4] are used, whichever were most favorable to the algorithms.

In order to make a fair comparison amonqst various algorithms, a similar final convergence accuracy should be obtained. This is especially so here, since the VO algorithm has a higher order of convergence than nreviously

proposed algorithms. In addition, thi availability of the hessian is used to detect saddles and local maxima. It was estimated that the results given for the existing algorithms were for a termination with maximum norm of the gradient less than 10^{-4}.

Table 3 summarizes the results of the comparisons. The table shows the number of iterations, the number of function evaluations, and the number of gradient evaluations for each algorithm including the VO algorithm. Except for the helical valley problem (30), the proposed algorithm is considerably better than the F algorithm, the algorithm judged by Himmelblau to be the best of all algorithms in the study. The helical valley function is not defined at the points $x=(0,0-,x_3)^T$, for any value of x_3, and therefore it is felt that these discontinuities affected the performance of the VO algorithm.

A second test on Wood's function (31) revealed another potential advantage of the VO algorithm. Implementations of the FR algorithm and of the DFP algorithm converged to the saddle point of this function when the initial guess was in a small neighborhood of the saddle point. The VO algorithm converged to the minimum point from the same initial guess. Thus, the VO algorithm may be more efficient in avoiding convergence to these troublesome points.

7. Conclusions

We have developed a new algorithm with a sound theoretical foundation for the minimization of a function of several variables. This algorithm performed well when compared with other popular algorithms. It has a novel scalar search which is in general along curved trajectories in the space of the independent variables. In addition, the algorithm has a high order of convergence, as high as four. The extension of the algorithm to the case when only function values, or function and gradient values can be supplied was entirely successful. The algorithm is more efficient however if the values of the hessian can also be supplied.

The computer program used in this research and a more detailed derivation with extensions of the proposed algorithm is given in [2].

REFERENCES

[I] D.G. Luenberger, Introduction to Linear and Nonlinear Programming, Reading, Mass., Addison-Wesley Publishing Co., 1973.

[2] A.J. Jimenez, "A Variable-Order Nonlinear Programming Algorithm for Use in Computer-Aided Circuit Design and Analysis," Ph.D. Dissertation, University of Florida, Gainesville, Florida, June 1976.

[3] W. Murray, "Second Derivative Methods," in Numerical Methods for Unconstrained Optimization, W. Murray (ed.), New York, Academic Press, 57-71, 1972.

[4] R.W.H. Sargent, and D.J. Sebastian, "Numerical Experience with Algorithms for Unconstrained Minimization*" in Numerical Methods for Non-linear Optimization, F.A. Lootsma (ed.), New York, Academic Press, 45-68, 1972.

[5] D.M. Himmelblau, Applied Nonlinear Programming, Mew York, McGraw-Hill Book Co., 1972.

[6] _____, "A Uniform Evaluation of Unconstrained Optimization Techniques," in Numerical Methods for Non-linear Optimization, F.A. Lootsma (ed.), New York, Academic Press, 69-97, 1972.

[7] H.H. Rosenbrock, "An Automatic Method for Finding the Greatest or the Least Value of a Function," Comp. J., 3, 175-184, 196-0.

[8] W.I. Zangwill, Nonlinear Programming: A Unified Approach, Englewood Cliffs, N.J., Prentice-Hall, Inc., 1969.

[9] O.L. Mangasarian, Nonlinear Programming, Mew York, McGraw-Hill Book Co., 1969.

[10] J.F. Traub, Iterative Methods for the Solution of Equations, Englewood Cliffs, N.J., Prentice Hall, Inc., 1964.

[II] J.M. Ortega, and W.C. Rheinboldt, Iterative Solution of Nonlinear Equations in Several Variables, New York, Academic Press, 1970.

REFERENCES

C.G. Broyden, "Quasi-Newton Methods," in Numerical Methods for Unconstrained Optimization, U. Murray (ed.), flew York, Academic Press, 87-106, 1972.

A.R. Colvi 11 e "A Comparative Study on Nonlinear Programing Codes," IBM N.Y. Sci. Center, Report 320-2949, 1963.

E.E. Cragg, and A.V. Levy, "Study *on* a Supennenory Gradient Method for the Minimization of Functions," J. Optim. Theory Appins., 4, 191-205, 1969.

J. Cullum, "Unconstrained Minimization of Functions without Explicit Use of Their Derivatives," IBM Research Report RC 3600, T.J. Watson Research Center, Yorktcwn Heights, N.Y., 1971 .

_____, "An Algorithm for Minimizing a -Differentiate Function that Uses Only Function Values," in Techniques *of* Optimization, A.V. fialakrishnan (ed.), New York, Academic Press, 117-127, 1972.

!J. Oavidon, "Variable Metric Methods for Minimization," A.E.C. Research and Development Rept. ANL-5990, Argonne National Lab., Argonne, 111., 1959.

_____, "Variance Algorithm for Minimization," Computer j., 10, 406-411, 1967.

R. Fletcher, "A New Approach to Variable Metric Algorithms," Conn. J., 13, No. 13, 317-322, 1970.

R. Fletcher, and M.J.D. Powell, "A Rapidly Convergent Descent Method for Minimization," Camp. J., 6, 163-168, 1963. Reprinted in Computer-Aided Circuit Design, S.W. Director (ed.), Stroudsburg, Perm., Dowden, Hutchinson and Ross, Inc., 361-366, 1973.

R. Fletcher, and CM. Reeves, "Function Minimization by Conjugate Gradients," Com;;. J. , 7, 149-154, 1964. Reprinted in Computer-Aided Circuit Design, S.W. Director (ed.), Stroudsburg, Penn., Dowden, Hutchinson and Ross, Inc., 367-372, 1973.

D.M. Kirrimel blau, Applied Nonlinear Programming, Nev; York, McGraw-Hill Book Co., 1972.

_____, "A Uniform Evaluation of Unconstrained Optimization Techniques," in Numerical Methods for Non-linear Optimization, F.A. Lootsma (ed.), Mew York, Academic Press, 69-97, 1972.

A.J. Jimenez, "A Variable-Order Nonlinear Programming Algorithm for Use in Computer-Aided Circuit Design and Analysis," Ph.D. Dissertation, University of Florida, Gainesville, Florida, June 1976.

D.G. Luenberger, Introduction to Linear and Nonlinear Programming, Reading, Mass., Addison-Wesley Publishing Co., 1973.

O.L. Mangasarian, Nonlinear Programming, New York, McGraw-Hill Book Co., 1969.

W. Murray, "Second Derivative Methods," in Numerical Methods for Unconstrained Optimization, W. Murray (ed.), New York, Academic Press, 57-71, 1972.

J.M. Ortega, and W.C. Rheinboldt, Iterative Solution of Nonlinear Equations in Several Variables, New York, Academic Press, 1970.

M.J.D. Powell, "An Efficient Method for Finding the Minimum of a Function of Several Variables Without Calculating Derivatives," Comput. J., 7, 155-162, 1964.

H.H. Rosenbrock, "An Automatic Method for Finding the Greatest or the Least Value of a Function," Comp. J., 3, 175-184, 1960.

R.W.H. Sargent, and D.J. Sebastian, "Numerical Experience with Algorithms for Unconstrained Minimization," in Numerical Methods for Non-linear Optimization, F.A. Lootsma (ed.), New York, Academic Press, 45-68, 1972.

G.W. Stewart, "A Modification of Davidon's Minimization Method to Accept Difference Approximations of Derivatives," J. ACM, 14, 72-83, 1967.

J.F. Traub, Iterative Methods for the Solution of Equations, Englewood Cliffs, N.J., Prentice-Hall, Inc., 1964.

W.I. Zangwill, Nonlinear Programming: A Unified Approach, Englewood Cliffs, N.J., Prentice-Hall, Inc., 1969.

TABLE 1 Results for Rosenbrock's problem with (a) the function, the gradient, and the hessian values supplied, (b) the function and the gradient values supplied, and (c) only the function values supplied. Note that the zero given for the eighth iteration in (a) was actually a printed result. The computer times required were .04 seconds for (a), .04 seconds for (b), and .053 seconds for (c) in an IBM/370 Model 165.

(a)

k	---Counters--- No, FUN	*no.* GRAD	No. HESS	ORDER	$\|\|x^k-x^*\|\|_\infty$	$f(x^k)-f(x^*)$	$\|\|f'(x^k)\|\|_\infty$
0	1	1	0	—	2.2	24.2	215.6
1	6	4	1	4	1.314	2.0921	12.1
2	11	7	2	4	.9343	1.55	15.25
3	18	10	3	4	.3961	.333	17.14
4	20	12	4	2	.3024	.0663	6.727
5	25	15	5	4	5×10^{-3}	$7.\text{£}\times10^{-3}$	3.545
6	30	18	6	4	1×10^{-3}	2.6×10^{-7}	1.3×10^{-3}
7	33	21	7	4	1×10^{-9}	7.3×10^{-20}	1.1×10^{-3}
8	34	22		2	0	0	0

TABLE 1 (continued)

(b)

k	--Counters— No. FUN	No. GRAD	ORDER	$\|x^k - x^*\|_\infty$	$f(x^k) - f(x^*)$	$\|\nabla f(x^k)\|_\infty$
0	1	1	—	2.2	24.2	215.6
1	3	6	4	1.314	2.092	12.1
2	15	11	4	.9845	1.549	15.23
3	24	16	4	.4091	.36	16.29
4	28	20	2	.3152	.06698	6.504
5	36	25	4	.0102	7.3×10^{-3}	3.411
6	43	33	4	4×10^{-3}	4.2×10^{-8}	3.6×10^{-4}
7	47	34	3	4×10^{-9}	3×10^{-18}	3.9×10^{-9}

(c)

k	Counters No. FUN	ORDER	$\|x^k - x^*\|_\infty$	$f(x^k) - f(x^*)$	$\|\nabla f(x^k)\|$
0	3	—	2.2	24.2	215.6
1	17	4	1.313	2.095	12.16
2	31	4	.9304	1.565	15.54
3	45	4	.1932	.5985	26.27-
4	59	4	.1219	1.3×10^{-2}	3.591
5	73	4	.02	5.2×10^{-4}	.4573
6	88	4	1×10^{-3}	3.6×10^{-7}	.0155
7	102	4	7×10^{-7}	$S.6 \times 10^{-U}$	1.6×10^{-8}
8	108	2	2×10^{-9}	4.8×10^{-19}	$5.5 \times in^{-10}$

TABLE 2 Summary of results for trie VO algorithm with STPEPS set to TO"[4]. The setting of MAXAV indicates, that the function, the gradient and the hessian values are supplied, if equal to 3; the function and the gradient values are supplied, if equal to 2; and only the function values are supplied if equal to 1. The point $\bar{x}$ is the convergence point.

Function	MAXAV	Counters Mo. ITN	f!o. FUN	No. GRAD	Me. HESS	$\|\bar{x}-x^*\|_\infty$	$f(\bar{x})-f(x^*)$	$\|f'(\bar{x})\|_\infty$
'(25)	3	7	32	20	7	5×10^{-S}	7×10^{-16}	7×10^{-8}
	2	7	46	33		1×10^{-7}	2×10^{-13}	2×10^{-5}
	1	7	94			6×10^{-6}	2×10^{-11}	9×10^{-5}
(29)	3	3	15	8	3	1×10^{-2}	8×10^{-8}	3×10^{-5}
	2	3	27	20		1×10^{-2}	8×10^{-6}	3×10^{-5}
	1	3	80			1×10^{-2}	7×10^{-S}	3×10^{-5}
(30)	3	9	46	26	9	5×10^{-7}	5×10^{-13}	9×10^{-6}
	2	10	75	57		5×10^{-6}	3×10^{-11}	6×10^{-5}
	1	10	202			1×10^{-6}	2×10^{-12}	2×10^{-5}
[3D	3	5	26	14	5	1×10^{-7}	2×10^{-14}	1×10^{-6}
	2	5	46	34		1×10^{-7}	1×10^{-14}	1×10^{-5}
	1	5	132			6×10^{-7}	1×10^{-11}	9×10^{-5}
(32)	3	6	26	16	6	5×10^{-2}	2×10^{-7}	3×10^{-5}
	2	4	38	28		3×10^{-2}	5×10^{-8}	1×10^{-5}
	1	4	111			5×10^{-2}	6×10^{-7}	9×10^{-5}

TABLE 3 Results of comparisons of the VO algorithm with seven other existing algorithms for the minimization of five standard test problems.

		Algorithms with supplied functions and gradients					Algorithms with only functions supplied			
Function	Counter	FR	DFP	B	F	VO	p	S	C	VO
(25)	No. ITN	27	19	35	39	7	37	23	25	7
	Mo. FUN	155	96	51	47	46	153	152	145	94
	i'lo. GRAD	28	20	36	47	33				
(29)	No. ITN	104	36	38	60	3	25	41	18	3
	No. FUN	624	434	374	68	27	966	622	148	80
	No. GRAD	105	37	39	68	20				
(30)	Mo. ITN	36	20	21	35	10	4	21	26	10
	Ho.. FUN	202	141	140	42	75	48	191	177	202
	No. GRAD	37	21	22	42	57				
(31)	No. ITN.	139	57	42	60	5	25	38		5
	No. FUN	3288	475	310	61	46	276	715		132
	Mo. GRAD	190	58	43	61	34				
(32)	No. ITN	39	96	84	82	4	36	128		4
	No. FUN	221	424	350	91	38	3480	1662		111
	No. GRAD	40	97	85	91	28				

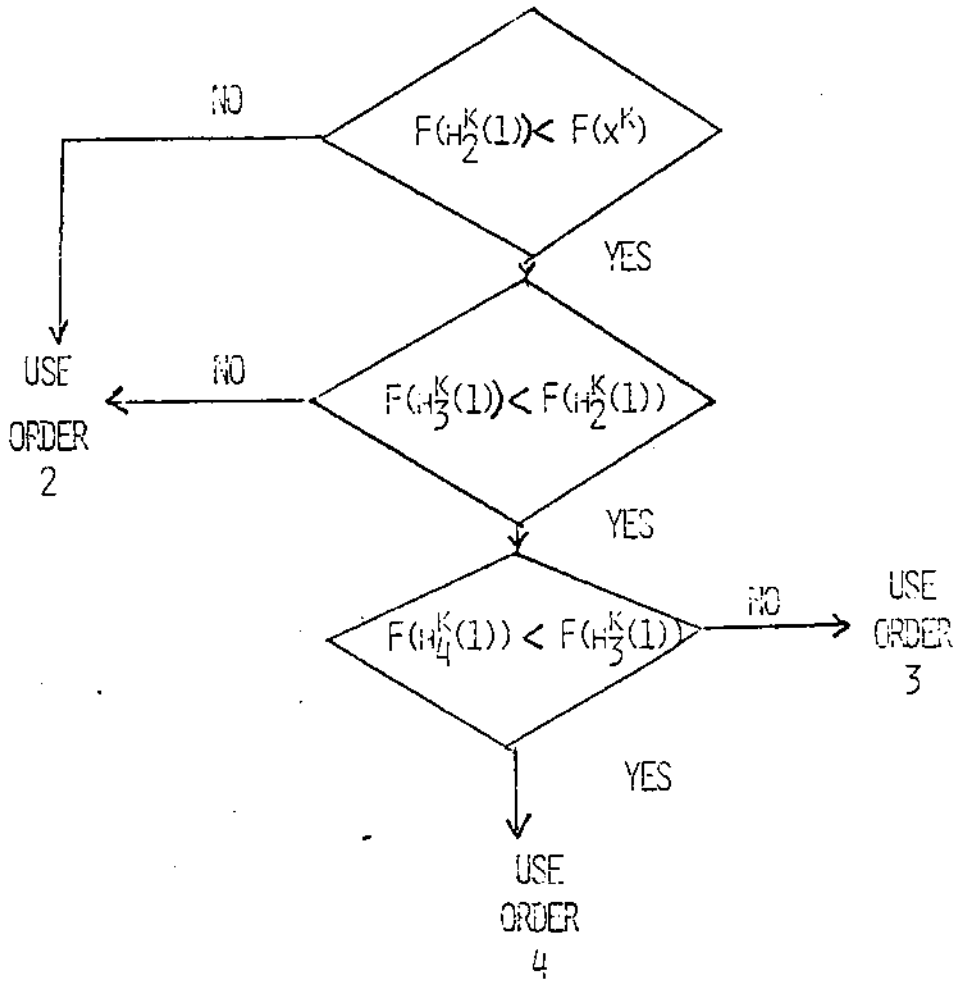


Figure 1. Flowchart of the strategy for selecting the transformation function order Tor the proposec! alçiori thin.

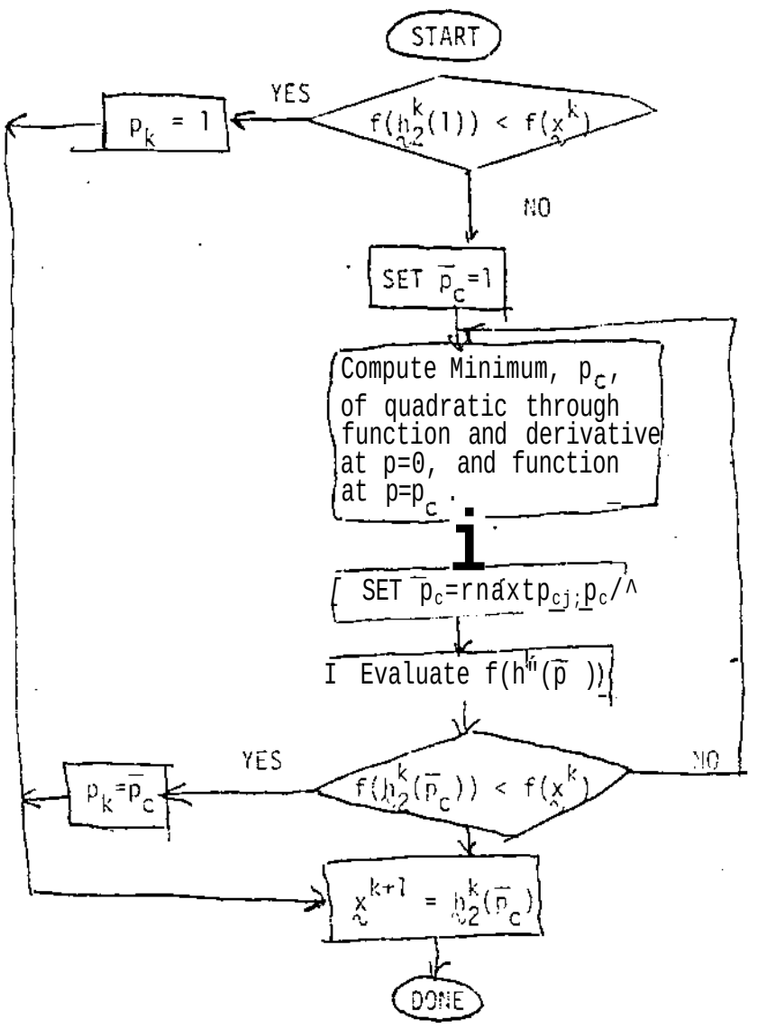
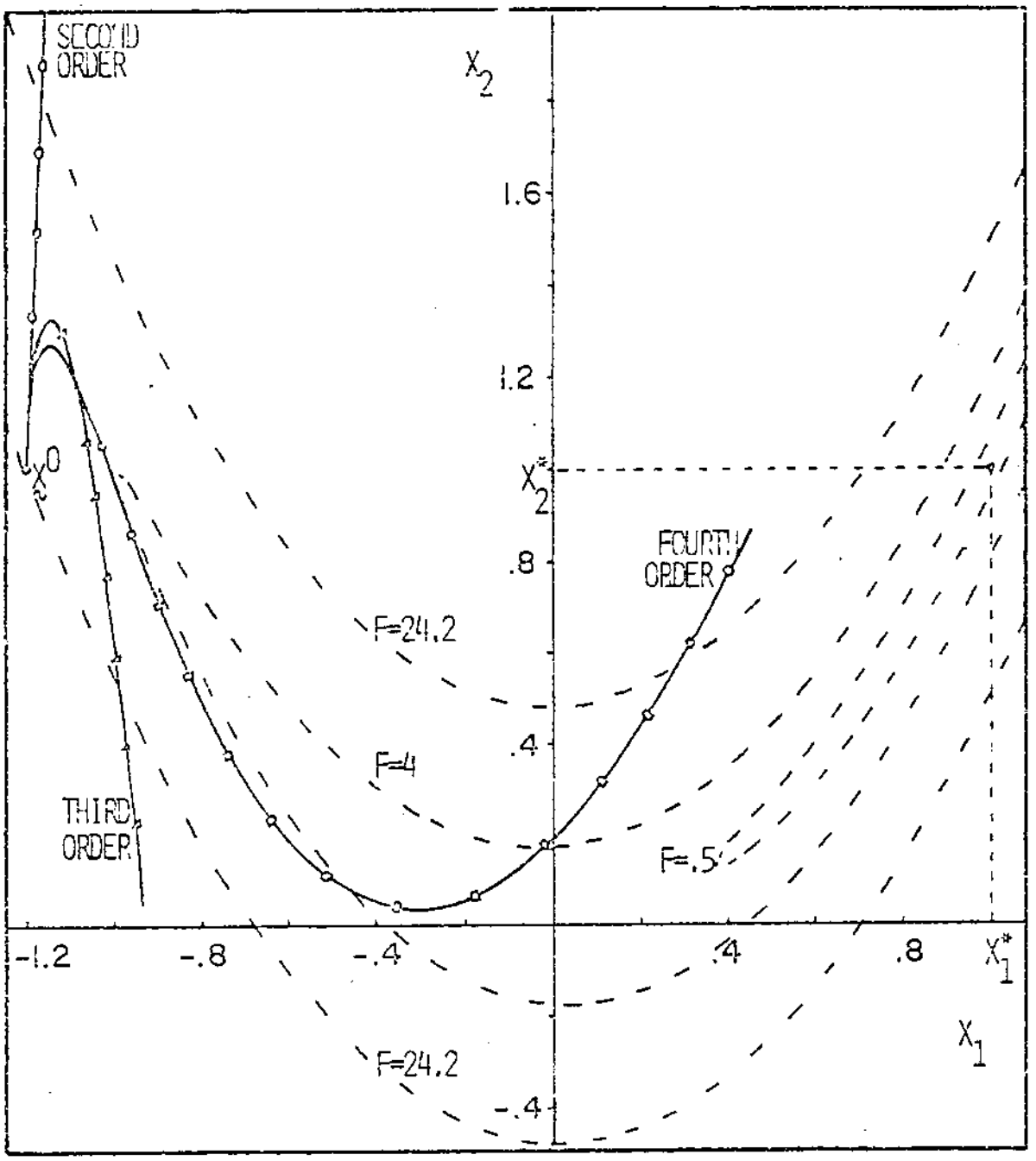


Figure 2. Scalar Search of the Algorithm when x is far from solution and when the second-order transformation is selected.



(a)

Fiyure 3 Trajectories for the second-, the third-, and the fourth-order transformation, (a) projected onto the $x_2, x^$ plane, (b) a three-dimensional view with "eye" at $x_1 = -1.3$, $x_2 = -.5$, f=0.

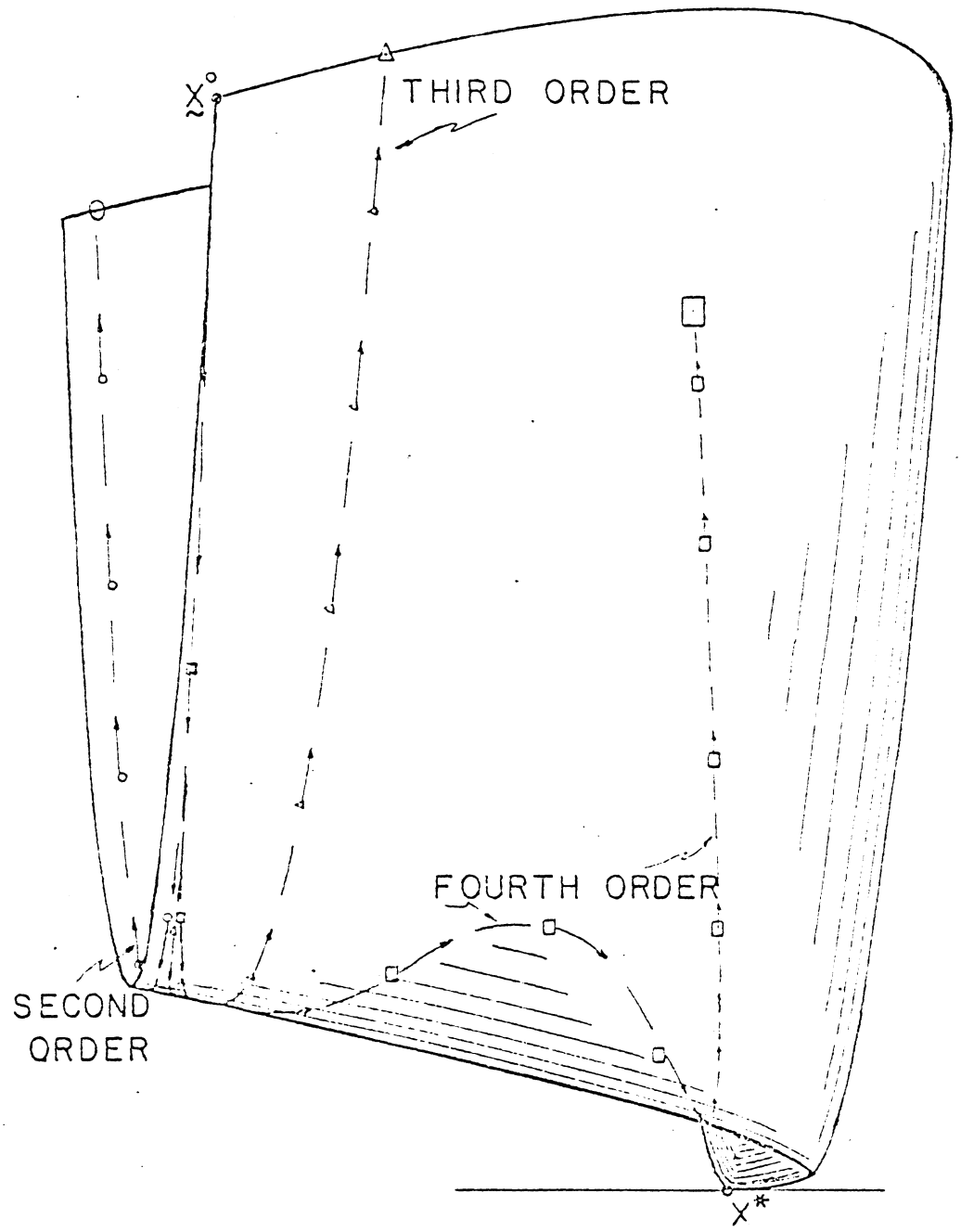


Figure 3b

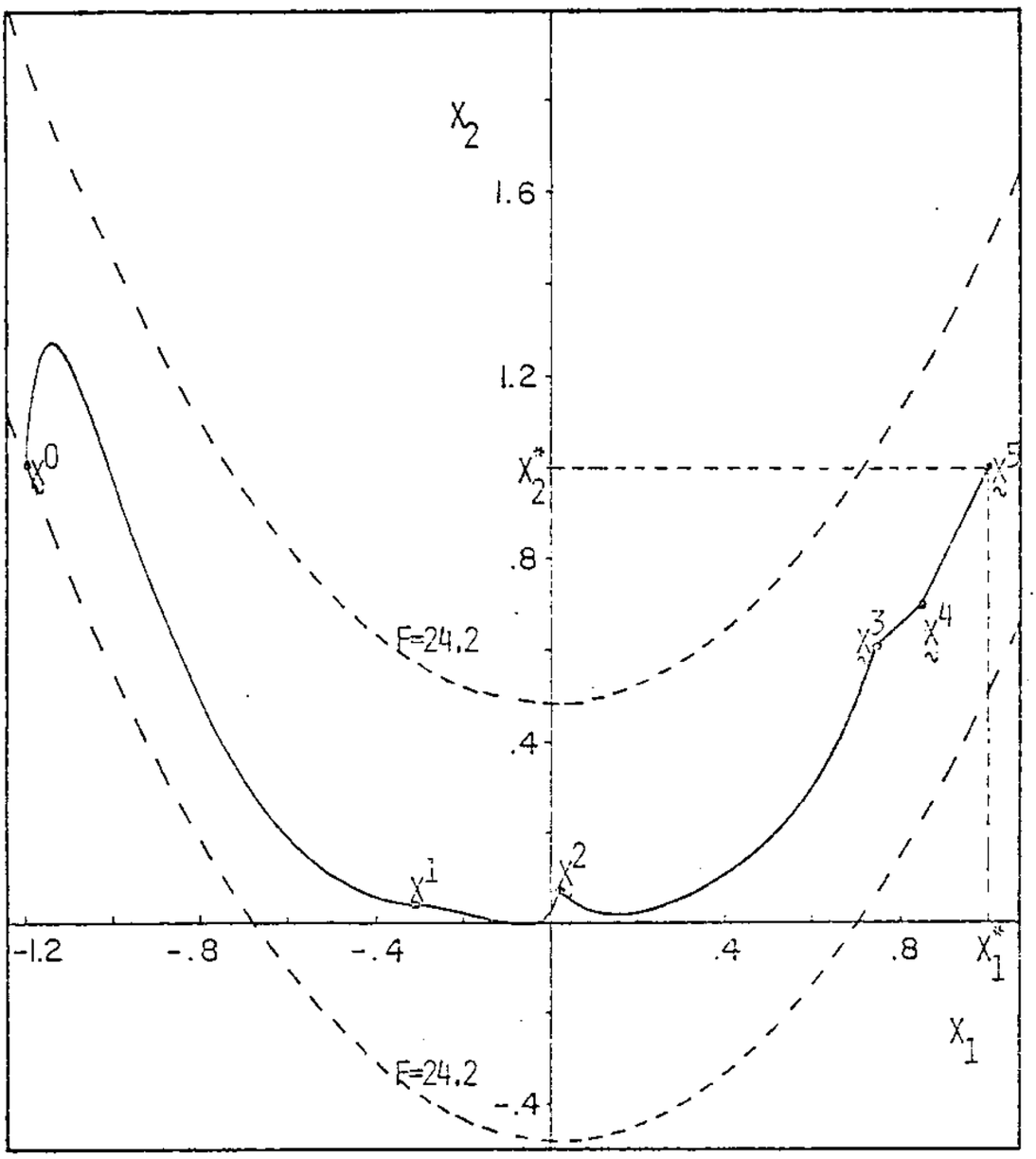


Figure 4 The First Five Iteration Trajectories for the Miniuization of Rosenbrock[1]s Functions.

FIGURE CAPTIONS

Figure 1. Flowchart of the strategy for selecting the transformation function order for the proposed algorithm.

Figure 2. Scalar search of the algorithm when x^k is far from solution and when the second-order transformation is selected.

Figure 3. Trajectories for the second-, the third-, and the fourth-order transformation, (a) projected onto the x_2?x_1 plane, (b) a three-dimensional view with "eye" at x-j = -1.3, x~$_2$ = -.5, f=0.

Figure 4. The first five iteration trajectories for the mnimization of Rosenbrock's functions.

[12] M.J.D. Powell, "An Efficient Method for Finding the Minimum of a Function of Several Variables without Calculating Derivatives," Comput. J., 7., 155-162, 1964.

[13] A.R. Colville, "A Comparative Study on Nonlinear Programming Codes," IBM N.Y. Sci. Center, Report 320-2949, 1968.

[14] E.E. Cragg, and A.V. Levy, "Study on a Supermemory Gradient Method for the Minimization of Functions," J. Optim. Theory Applns., 4, 191-205, 1969.

[15] R. Fletcher, and M.J.D. Powell', "A Rapidly Convergent Descent Method for Minimization," Comp. J., 6, 163-168, 1963. Reprinted in Computer-Aided Circuit Design, S.W. Director (ed.), Stroudsburg, Penn., Dowden, Hutchinson, and Ross, Inc., 361-366, 1973.

[16] W. Davidon, "Variable Metric Methods for Minimization,[11] A.E.C. Research and Development Rept. AiJL-5990, Argonne National Lab., Argonne, 111., 1959 and "Variance Algorithm for Minimization," Computer J., 10, 406-411, 1967.

[17] R. Fletcher, and M.J.D. Powell, "A Rapidly Convergent Descent Method for Minimization," Comp. J., 6, 163-168, 1963. Reprinted in Computer-Aided Circuit Design, S.W. Director, (ed.), Stroudsburg, Penn., Dowden, Hutchinson and Ross, Inc., 361-366, 1973.

[18] C.G. Broyden, "Quasi-:!ewton Methods," in Numerical Methods for Unconstrained Optimization, W. Murray (ed.), *iew York, Academic Press, 87-106, 1972.

[19] R. Fletcher, "A New Approach to Variable Metric Algorithms," Comp. J., 13, Mo. 13, 317-322, 1970.

[20] G.W. Stewart, "A Modification of Davidon's Minimization Method to Accept Difference Approximations of Derivatives," J. ACM, 14, 72-83, 1957.

[21] J. Cull urn, "Unconstrained Minimization of Functions without Explicit Use of Their Derivatives," IBM Research Report RC 3600, T.J. Watson Research Center, Yorktown Heights, M.Y., 1971.

[22] J. Culium, "An Algorithm for Minimizing a Differentiate Function that Uses Only Function Values," in Techniques of Optimization, A.V. Balakrishnan (ed.), Mew York, Academic Press, 117-127, 1972.

FOOTNOTES

1. Other functions can also be used which yield different series expansions, one of which is the n-dimensional extension of a previously known series attributed to Euler, [2].
2. z_i^p denotes z_i raised to the p^{th} power.
3. In addition it was experimentally found that for one function tested, transformation functions of order greater than four did not increase efficiency.
4. See also [5] and [6].