"Direct Search" Solution of Numerical and Statistical Problems*

ROBERT HOOKE AND T. A. JEEVES

Westinghouse Research Laboratories, Pittsburgh, Pennsylvania

1. *Introduction*

In dealing with numerical problems for which classical methods of solution are unfeasible, many people have tried various procedures of searching for an answer on a computer. Our efforts in this direction have produced procedures which seem to have had (for us and for others who have used them) more success than has been achieved elsewhere, so that we have been encouraged to publish this report of our studies.

We use the phrase "direct search" to describe sequential examination of trial solutions involving comparison of each trial solution with the "best" obtained up to that time together with a strategy for determining (as a function of earlier results) what the next trial solution will be. The phrase implies our preference, based on experience, for straightforward search strategies which employ no techniques of classical analysis except where there is a demonstrable advantage in doing so.

We have found it worthwhile to study direct search methods for the following reasons:

(a) They have provided solutions to some problems, of importance to us, which had been unsuccessfully attacked by classical methods. (Examples are given below.)

(b) They promise to provide faster solutions for some problems that are solvable by classical methods. (For example, a method for solving systems of linear equations, proposed in Section 5, seems to take an amount of time that is proportional only to the first power of the number of equations.)

(c) They are well adapted to use on electronic computers, since they tend to use repeated identical arithmetic operations with a simple logic. Classical methods, developed for human use, often stress minimization of arithmetic by increased sophistication of logic, a goal which may not be desirable when a computer is to be used.

(d) They provide an approximate solution, improving all the while, at all stages of the calculation. This feature can be important when a tentative solution is needed before the calculations are completed.

(e) They require (or permit) different kinds of assumptions about the functions involved in various problems, and thus suggest new classifications of functions which may repay study.

Direct search is described roughly in Section 2, and explained heuristically in Section 3. Section 4 describes a kind of strategy. Sections 5 and 6 describe

* Received May, 1960; revised October, 1960.

examples in which we have used direct search in curve fitting problems, solving integral equations, maximizing or minimizing functions with or without restrictions on the independent variables, and solving systems of equations.

2. *Simple Description of Direct Search*

The application of direct search to a problem requires a space of points P which represent possible solutions, together with a means of saying that P_1 is a "better" solution than P_2 (written $P_1 \subset P_2$) for any two points in the space. There is presumably a single point P^*, the solution, with the property $P^* \subset P$ for all $P \neq P^*$.

In these terms, the basic form of direct search is as follows. A point B_0 is arbitrarily selected to be the first "base point". A second point, P_1, is chosen and compared with B_0. If $P_1 \subset P_0$, P_1 becomes the second base point, B_1; if not, B_1 is the same as B_0. This process continues, each new point being compared with the current base point. The "strategy" for selecting new trial points is determined by a set of "states" which provide the memory. The number of states is finite. There is an arbitrary initial state S_0, and a final state which stops the search. The other states represent various conditions which arise as a function of the results of the trials made. The kind of strategy used is dictated by various aspects of the problem, including one's knowledge of the structure of the solution space. The strategy itself comprises the choice of B_0 and S_0, the rules of transition between states, and the rules for selecting trial points as a function of current state and base point.

Suppose, for example, that the problem is to minimize a function $f(x_1, x_2, \cdots, x_n)$. A solution point P_i is a vector $(x_{1i}, x_{2i}, \cdots, x_{ni})$, and we say that $P_i \subset P_j$ if and only if

$$f(x_{1i}, x_{2i}, \cdots, x_{ni}) < f(x_{1j}, x_{2j}, \cdots, x_{nj}).$$

The base point B_r, then, is simply that point, from among $B_0, P_1, P_2, \cdots, P_r$, which has produced (apart from ties) the smallest value of $f(x_1, x_2, \cdots, x_n)$. The next trial point, P_{r+1}, is determined (relative to B_r) by the present state S_r.

It is convenient to think of a trial at P_{r+1} as a "move" or "step" from the base point B_r. The move is a "success" if $P_{r+1} \subset B_r$, and is a "failure" otherwise. Roughly speaking, the states make up part of the logic, influencing moves to be proposed in the same general direction (assuming that direction is meaningful in the solution space) as those which have recently succeeded; they suggest new directions if recent moves have failed; and, finally, they decide when no further progress can be made. The fact that no further progress can be made does not always indicate that the solution has been found. Thus direct search may fail. Since, so far, no useful sufficient conditions for the success of direct search have been derived, we recommend the method for the following types of problem:

(a) problems for which the answers can be tested, once found;

(b) problems which consist of many separate cases, a few of which can be checked by alternative methods.

In addition, if neither situation holds, a partial check can be obtained by using direct search several times, with different (perhaps randomly chosen) starting points each time. This makes direct search available for problems on which all other methods fail.

A more precise definition of the basic form of direct search is given in Appendix A. Various modifications, of course, may be desirable in special cases. The pattern search described in Section 4 and Appendix B is such a modification.

3. *Heuristic Justifications*

In discussing direct search with others, we have found that certain questions are repeatedly asked. It seems in order to comment on such questions at this point. The questions fall naturally into two groups:

(A)

1. Under what conditions does direct search converge to the solution?
2. Once an approximate solution is found, how good is it?
3. What is the effect of error, if any, in the functional determinations?

(B)

1. What are the advantages, if any, over classical methods, such as polynomial approximations?
2. How can one justify neglecting information? (For example, referring to the preceding section, we often know, not only a way of comparing P_r and B_{r-1}, but also a way of measuring the difference. If so, why is this difference not always used?)

The questions in group A are pertinent and suggest areas of research. We have no answers to them except to point out that, for many classical procedures, the same questions are still unanswered. Also, such answers as have been given are often of more academic than practical interest. For example, the fact that a given function is the limit of a convergent sequence of polynomials is in general neither necessary nor sufficient to the useful approximation of that function by a polynomial.

The questions in group B can be discussed in terms of the optimization problem (that of finding the maximum or minimum of a function of several variables). It has become orthodox to solve such problems by using some form of the "method of steepest ascent," abbreviated MSA below. The remarks below are not intended to be critical of this useful method, but rather to recall its limitations, well known to many, but apparently not all, of its users.

(a) The only justification of the MSA is an intuitive one, based on such physical interpretations as blind men climbing mountains or rivers finding their way to the ocean. In numerical problems one is not restricted to continuous search, and it is hard to see why these particular analogies should recommend it. There appears to be no reason why proposed alternatives should have to justify themselves in terms of the MSA.

(b) An obvious way of attacking the optimization problem is to make some observations, fit a second-degree polynomial, find the maximum of the quadratic and use this point as a location for iteration of the attack. Experience has shown

that the returns from this activity are not commensurate with the effort required. The amount of effort expended locally has therefore been reduced by fitting only a first-degree polynomial, and this simplified attack is the MSA. Direct search carries the simplification one step farther. It is clear even if one looks at the simplest non-trivial case (a quadratic function of two variables, having non-circular contours) that the direction of steepest ascent is only loosely associated with the direction to the maximum point, so that work done in finding it precisely hardly seems worthwhile. The fact that the direction of steepest ascent is not invariant with respect to changes of scale is further evidence of its unimportance. Experience with direct search methods indicates that the MSA computations represent an inefficient use of computation time.

(c) The MSA is not, as some seem to think, an objective method. Once the direction of steepest ascent has been determined, the MSA itself provides no means of searching along this direction.

In short, our answer to the questions in group B is that there is no virtue in the manipulation of information merely because it is available. The burden of justification should fall at least as much on those who spend time processing information as on those who do not use it. Local information provides rough guideposts, but exacting examination of it is of dubious value. As the search proceeds from locality to "better" locality, information acquired in earlier stages becomes less and less useful and should be discarded as soon as it is obsolete.

The reader may share the feelings of the authors that this section has been rich in opinions but somewhat deficient in facts. We attribute this to the state of the field of numerical analysis, which provides little by way of a theoretical framework for the comparison of numerical methods. The test of a numerical method must usually be empirical, and in the remaining sections we therefore provide some examples on which we have used direct search methods.

4. *Pattern Search—A Specific Kind of Strategy*

Pattern search is a direct search routine for minimizing a function $S(\varphi)$ of several variables $\varphi = (\varphi_1, \varphi_2, \cdots, \varphi_K)$. The argument φ is varied until the minimum of $S(\varphi)$ is obtained. The pattern search routine determines the sequence of values for φ; an independent routine computes the functional values of $S(\varphi)$.

The operative mechanism of the routine will now be indicated. The successive values of φ can be interpreted as points in a K-dimensional space. The procedure of going from a given point to the following point is called a *move*. A move is termed a *success* if the value of $S(\varphi)$ decreases; otherwise, it is a *failure*. The pattern search routine makes two types of move. The first type of move is an exploratory move designed to acquire knowledge concerning the behavior of the function $S(\varphi)$. This knowledge is inferred entirely from the success or failure of the exploratory moves without regard to any quantitative appraisal of the functional values. The rudimentary information of success or failure is utilized

by combining it into a "pattern" which indicates a probable direction for a successful move. The exploratory moves form a (vector) basis for the argument space. For simplicity, the exploratory moves are here taken to be *simple*, that is, at each move only the value of a single coordinate is changed. The second type of move is a pattern move designed to utilize the information acquired in the exploratory moves, and accomplish the actual minimization of the function by moving in the direction of the established "pattern". As set up each pattern move is followed by a sequence of exploratory moves which continually revise the pattern. The point from which a pattern move is made is designated a *base point*, and the direct search procedure may be conceived as fundamentally proceeding from base point to base point.

Although the deduction of "optimum" properties for direct search procedures has proved difficult, intuitive arguments for pattern search can be given. The pattern move from a given base point duplicates the combined moves from the previous base point. That is, all coordinates are changed by an amount equal to the difference between the present base point and the previous base point. The intuitive basis for this type of move is the presumption that whatever constituted a successful set of moves in the past is likely again to prove successful. The result of the pattern move may be either a success or a failure. The consequent routines are discussed separately below.

Following a successful pattern move it is reasonable to conduct a series of exploratory moves and attempt to further improve the result. Each exploratory move is carried out as follows: A single coordinate of the point is varied to see whether a successful move can be made by either increasing or decreasing this coordinate by a prescribed step size. If a success is obtained the altered value of the coordinate is retained; otherwise, the original value is restored. Such exploratory moves are made for each coordinate, and the final point reached becomes a new base point. In this way the successful pattern move is further improved and only the information as to whether the exploratory moves succeed or fail is used.

If the pattern move fails, the simplest way of continuing the search would be to begin over again from the base point with a series of exploratory moves, and thus establish an entirely new pattern. Experience has shown that this continuation procedure is basically sound. Typically a pattern once established will, through continuous modification, grow until the length of the pattern move is 10 to 100 times the basic step size. When the pattern move then fails, it is generally not possible to make any further significant progress in a direction similar to that established by the pattern, and no simple modification of the pattern permits a good new direction to be selected.

An exception to the above argument must be made when the pattern is first being established. Also, it does happen that previously successful patterns can sometimes again be made successful by slight modification. In these two cases, essentially similar patterns would be produced by starting the search over again. Since progress towards the goal of minimizing the function is only really made in the pattern moves, it is desirable to reduce the number of exploratory moves,

and to retain patterns when possible. For these reasons, it has proved highly desirable to also follow a pattern move which fails by a series of exploratory moves. The final point then reached is compared with the base point of the pattern move, and if the overall set of moves is now a success this final point becomes the new base point. Further efforts to retain an established pattern have not been deemed worthwhile. Consequently, if the overall set of moves is a failure the pattern is abandoned and the search begun again from the base point. (Note that with the addition of the procedure of this paragraph the success or failure of the pattern move becomes irrelevant, since in either case the pattern move is followed by exploratory moves.)

The direct search procedure outlined above has been termed pattern search because it is based on the determination of a "pattern" of simple moves that will give a successful direction in which to move. From the above description it can be realized that the major reduction in $S(\varphi)$ is produced by the pattern move. Some reduction is made by the exploratory moves, but their primary function is to supply information for the improvement of the pattern move.

In practice, pattern search has proved particularly successful in locating minima on hypersurfaces which contain "sharp valleys". On such surfaces classical techniques behave badly and can only be induced to approach the minimum slowly. Direct search procedures using only simple moves are forced into small step sizes in order to keep from moving out of the valley on each step. Consequently, though faster than classical techniques, such direct search procedures are not overly fast. Pattern search has the inherent potentiality of making pattern moves directly down the valley, and hence rapidly approaching the minimum.

The method of terminating the search has not been mentioned. For any given value of the step size, the search procedure will reach an impasse when, the pattern move having failed to determine a new base point, all the exploratory moves from the base point fail. To further continue the search, it is necessary to reduce the prescribed step size. The amount of reduction should be enough to permit a new pattern to be established. However, too large a reduction in step size will result in slowing down the search. In practice, the total search time has not proved to be overly sensitive to the amount of reduction. The final termination of the search is made when the step size is sufficiently small to insure that the optimum has been closely approximated. In any case, the step size must be kept above a practical limit imposed by the means of computation.

An exact description of a pattern search routine, which has been applied, is given in Appendix B.

5. *Some Examples*

In this section we shall discuss first two examples of curve fitting in which we used direct search when other methods failed, and next the problem of solving systems of linear equations which we have studied in order to compare direct search with conventional methods where the latter are most useful. All

of these problems are attacked by turning them into optimization problems, so there is no need to consider a separate example of these. The optimization problem with restricted variables is discussed in the next section.

The first curve-fitting problem came to us as a set of measurements of neutron flux in a nuclear reactor, to be fitted by the curve

$$y(x) = A \cos (Bx + E) + C \cosh (Dx + E).$$

A set of "adjusted" counts y_i were made at points x_i, $i = 1, 2, \cdots, n$. The variance of each count was proportional to its mean. The fit of the curve was therefore judged by the weighted sum of squares

$$S = \sum_{i=1}^{n} \frac{[y_i - A \cos (Bx_i + E) - C \cosh (Dx_i + E)]^2}{a_i [A \cos (Bx_i + E) + C \cosh (Dx_i + E)]},$$

so that the problem was to find A, B, C, D, and E to minimize S. (The a_i are factors related to the manner in which the counts were adjusted.)

The task of minimizing S by classical methods is a formidable one, and it was decided to simplify the problem by minimizing

$$S^* = \sum_{i=1}^{n} (a_i y_i)^{-1} [y_i - A \cos (Bx_i + E) - C \cosh (Dx_i + E)]^2.$$

(Since the errors $y_i - x_i$ were small compared to the magnitude of $y(x_i)$, it was reasonable to replace the denominator in S by the approximation $a_i y_i$.)

To handle the non-linear parameters B, D, and E, the usual linearization device was used. The function S^* was expanded into a power series, through quadratic terms, about a first estimate of the parameter values. The parameter values minimizing this quadratic surface were expected to afford a better estimate of the parameter values which minimize S^*. By iterating this expansion and minimization process, it was anticipated that a convergent sequence of estimates would be obtained, provided only that the initial estimates were not too bad.

For this problem, very close approximations could easily be made to A, B, and E, while good approximations to C and D were constructible. It was expected therefore that a few iterations on a computer would supply a highly precise solution. The first application of the process, however, yielded a set of second approximations with a much larger value of S^*, and the process had failed.

An auxiliary technique, intended to force the reduction of S^*, was devised to operate with a minimum of additional information. Using only the value of S^* for the new parameters, the third derivative in the direction of the parameter change was estimated, and re-evaluation of the minimum was made by fitting a cubic curve in this direction. This seemed sufficient to insure convergence, but it turned out to be discouragingly slow. The source of the difficulty was that, for the arguments considered, the exponential character of the "cosh" term prevented reasonable approximation by a quadratic function.

The inadequacy of the classical approach was further emphasized by the fact

that an intelligent person could obtain much better estimates in less time without using all these derivatives. Since the classical procedure required the calculation of 20 derivatives and 2 functional evaluations for each iteration, the question arose as to whether a direct search procedure could make greater progress in 22 steps (i.e., 22 functional evaluations) than the classical procedure could make in each iteration. When a direct search strategy (though a very simple one) was used, its progress was so much greater that the two methods could not sensibly be compared.

The direct search method was also applied to the more difficult problem of minimizing S rather than S^*, and a fairly simple strategy produced a satisfactory solution in less than an hour of medium-speed computer time. Since then, other results have indicated that this time can be reduced by at least a factor of 10 by minimizing S^* instead of S, using an incremental procedure (Appendix C) to shorten the functional computations, and by using pattern search.

A similar problem, connected with some metallurgical studies, involved the fitting of a curve

$$\rho(T) = a + bT + cT^3 - dT \exp(-q/T).$$

The non-linearity introduced by q, and an additional condition requiring that a, b, d, and q be non-negative, made the problem unsolvable by usual techniques. A simple direct search routine, used with a standard computational subroutine for the exponential, turned out to be too slow. An incremental procedure (taking advantage of the fact that move sizes are constant) was substituted for the functional computation, reducing the time on the medium-speed computer to slightly over an hour per problem. It is expected that, if further problems of this sort come to us, the combination of incremental functional computation with the pattern search strategy will reduce the machine time to a few minutes.

It is clear that direct search can be used to solve a system of one or more equations of the form

$$f_i(x_1, x_2, \cdots, x_n) = 0, \qquad i = 1, 2, \cdots, n$$

where the form of the f_i is practically arbitrary, so long as the solution is unique. One has only to minimize the function

$$\sum_{i=1}^{n} [f_i(x_1, x_2, \cdots, x_n)]^2.$$

Rather than invent difficult equations to solve, it was decided to try direct search on systems of linear equations, represented by

$$Ax = b, \tag{1}$$

where A is an $n \times n$ matrix and x and b are vectors. We let u stand for a trial solution of the system and let $\epsilon = u - x$. It was found convenient to solve the problem by minimizing the following, where the superscript T denotes the transpose:

$$\epsilon^T B \epsilon.$$

Here B is a positive definite matrix equal to A (if A is positive definite). Since x is unknown, the value of $\epsilon^T A \epsilon$ is also unknown; however, the change in $\epsilon^T A \epsilon$ with a change in u is computable, and this information is sufficient to allow minimization by direct search. (If A is not positive definite the equivalent set of equations $A^T A x = A^T b$ are solved and B is taken equal to the positive definite matrix $A^T A$.)

A preliminary evaluation of the direct search method was made by solving (1) with the positive definite matrix $A = \alpha I + J$ being selected for experimentation. Here I is the unit matrix and J is the matrix all of whose elements are ones. By varying α the condition number P of the matrix A could be varied to yield a well-conditioned matrix (P small) or an ill-conditioned matrix (P large). The condition number P of a matrix is defined to be the ratio of the largest to the smallest characteristic root. In a general sense, the condition number measures the difficulty in solving the system. The right-hand side b was taken simply to be the sequence of odd numbers. Table I gives the times required to obtain 5 significant figures in a solution. The times quoted are for the direct search process alone and do not include the time, of about n seconds, for initial scaling.

In order to explain the results of the use of direct search on systems such as (1), the effect of an ill-conditioned matrix A must be described. In classical procedures, a sufficiently ill-conditioned matrix may cause the solution obtained to be meaningless, but it has no effect on the time required. With direct search, an ill-conditioned matrix causes the computation time to increase; there is hope, however, that the solution is ultimately obtained.

For comparison, two conventional programs were applied to the same equations (See Table II. Note that the natures of the different methods preclude any absolute comparison.) The most interesting result is the apparent linear increase with n of the time of solution using the direct search method. This is in contrast with the well known fact that for conventional methods the time of solution is proportional to n^3. This result, which could be anticipated from the principles of the method, points to direct search as a technique suited to solving large sets of linear equations.

TABLE I

Direct Search Method—Time for Solution in Seconds

	$P = 2$	$P = 11$	$P = 101$
$n = 5$	13	21	161
$n = 10$	27	30	277
$n = 15$	47	63	209
$n = 20$	70	95	413

TABLE II

Conventional Methods—Time for Solution in Seconds

($n = 20$ only)

Datatron Floating Point Program (6 significant figures)	132 seconds
Datatron Fixed Point Program (Modified) (7 significant figures)	270 seconds

Certain other features of the direct search method can be observed by examining Table I. Firstly, the time for solution is a stochastic (random) variable, and the actual time depends on chance effects occurring during the course of the search. Secondly, the time for solution depends on the condition of the matrix, and as the matrix becomes progressively more ill-conditioned the time for solution increases. This deleterious effect of the condition has been partially obviated by the use of pattern search. This method has been developed through study of the behavior of the simple direct search method used to obtain the above results. Pattern search is well adapted to cope with the geometrical situation characteristic of ill-conditioned matrices. It has been applied to a particular 4-variable matrix, chosen because it was very ill-conditioned and had given difficulty under previous classical methods of solution. Whereas the simple direct search procedure could not obtain a solution in a reasonable time, pattern search was able to develop the solution in 60 to 90 seconds per significant digit. Experience with this early version of pattern search has been incorporated in the improved version given in Section 4 and Appendix B. Various modifications of pattern search are now under study to determine their behavior with ill-conditioned matrices. It is conjectured that the use of a strategy of pattern search type will make the direct search method a very useful one in the solution of very badly conditioned systems, as well as systems of large size.

6. *Problems Whose Solutions Are Functions*

In this section we consider problems which require the determination of a function satisfying certain conditions. (For such problems, numerical approximations generally provide solutions which are expressible as a finite collection of numbers. Strictly speaking, then, there is no difference between a problem whose solution is a set of numbers and one whose solution is a set of functions, but the point of view is different and we find the dichotomy useful.)

We are concerned here with the type of problem for which there is defined a family of functions that are permissible solutions, the solution being that member of the family which most nearly satisfies the conditions stated in the problem. The case that has been treated most is of course that in which the family of permissible solutions is defined by some elementary function containing one or more parameters that vary over some domain. If these parameters enter linearly, solutions can be obtained by classical methods; if they enter nonlinearly, one can use least squares with direct search, as exemplified in Section 5.

The case of interest here is that in which the family of permissible solutions cannot be described in terms of a function known except for some parameters. Often an experimenter, wishing to fit a curve to some data, knows nothing more than the general shape of the solution. For example, in dealing with a function of one variable, he may believe (from theory or experience) that the function is monotone, or convex, or something of this sort. Prior knowledge of this kind is difficult, if not impossible, to use in conjunction with classical methods.

A problem that we have solved by direct search methods is the following. A

situation described by the equation

$$f(x) = \int_{-\infty}^{\infty} \varphi(u)k(x - u)\, dx$$

had been studied experimentally to the extent that $f(x)$ and $k(y)$ had been observed, with error, at a finite number of points. The problem was to deduce $\varphi(u)$, known, by physical considerations, to be unique.

At first, a step-function solution was tried. This turned out to require too much labor unless the successive approximations could be performed by a computer. Next, polynomial approximations were tried. It was clear that the degree of the polynomial would have to be fairly high, say 8 or 10. Measurement error, however, caused the fitted polynomial curves to have an absurd (from the physical point of view) number of bends. The same was true of a solution in which $\varphi(u)$ was approximated by polygons.

Finally, it became clear that the known properties of $\varphi(u)$ would have to be incorporated into the solution from the beginning. These were:

$$\begin{aligned}
&\varphi(u) = 0, && u \leqq a \text{ for some } a\\
&\varphi(u) \geqq 0, && u > a\\
&\varphi'(u) \geqq 0, && u < b \text{ for some } b\\
&\varphi'(u) \leqq 0, && u > b\\
&\varphi''(u) \geqq 0, && u < c \text{ for some } c\\
&\varphi''(u) \leqq 0, && c < u < d \text{ for some } d\\
&\varphi''(u) \geqq 0, && d < u
\end{aligned}$$

The problem was then solved, to the satisfaction of the physicists who presented it, by representing $\varphi(u)$ by a polygon whose vertices satisfied inequalities on differences corresponding to the inequalities on derivatives given above. The family of permissible solutions consisted of polygons (each determined by 20 vertices). The problem was treated as an optimization problem in 20 variables (the vertices of the polygonal approximations), the idea being to minimize a certain error function which represented the degree of failure of a certain approximation to $\varphi(u)$ to satisfy the original integral equation.

A simpler problem, conceptually, is that of simply fitting a curve to a set of points obtained empirically. A scientist, confronted with this problem, and having no theoretical form for the relationship under study, usually resorts to "fitting a smooth curve by eye." If one could find out what he means by a "smooth curve", it could probably be expressed as one of a family of curves, several derivatives of which satisfy certain inequalities. If one could provide him with a numerical method for solving his problem when put in such a form, he might be willing to learn how to do it, and thus add a degree of objectivity to this very prevalent kind of scientific inference.

To show how direct search might provide such a numerical method, let us consider the following very simple problem. Given the data points (a_i, b_i) $i =$

$1, 2, \cdots, n$, find the "best" fit $y = f(x)$, given only that

$$f'(x) \geqq 0$$

$$f''(x) \leqq 0.$$

We reformulate the problem as follows: Find the values y_i, $i = 1, 2, \cdots, n$, to minimize

$$S = \sum_{i=1}^{n} (y_i - b_i)^2$$

subject to the restrictions

$$y_{i+1} - y_i \geqq 0, \qquad i = 1, 2, \cdots, n-1,$$

$$y_{i+2} - 2y_{i+1} + y_i \leqq 0, \qquad i = 1, 2, \cdots, n-2.$$

This is now an optimization problem in n variables and can be attacked by direct search. In addition to the usual difficulties there is the fact that the solution must lie on the boundary of a rather complicated region. Some strategies that had succeeded on simpler problems were tried without success, and so the following attack was devised.

Let

$$f(x_1, x_2, \cdots, x_n) \tag{2}$$

be a function to be minimized, subject to the restrictions

$$\begin{aligned} g_1(x_1, x_2, \cdots, x_n) &\leqq 0 \\ &\vdots \\ g_r(x_1, x_2, \cdots, x_n) &\leqq 0. \end{aligned} \tag{3}$$

Then, instead of searching for an answer over the complicated "permissible region" defined by (3), we proceed to minimize the function

$$f(x_1, \cdots, x_n) + \sum_{i=1}^{r} k_i u_i$$

over the whole n-dimensional space, where

$$u_i = \begin{cases} g_i(x_1, x_2, \cdots, x_n) \text{ if this is positive} \\ 0 \quad \text{otherwise,} \end{cases}$$

$$\text{and} \quad k_i \geqq 0, \qquad i = 1, 2, \cdots, r.$$

Thus $k_i u_i$ represents a "penalty" for not satisfying the inequality

$$g_i(x_1, \cdots, x_n) \leqq 0.$$

For a given set of k's a solution can be found, and as the k's are increased this solution should approach the solution desired.

This method has been tried out on simple examples, with encouraging results. It hardly needs to be pointed out that much remains to be learned about this method. There will be times when it does not produce a satisfactory solution.

The solution itself is difficult to interpret, particularly with regard to its variability in the face of error. Nevertheless, it is to us a step in an interesting direction, and one which may turn out to be profitably used on some of the many problems of maximizing or minimizing over a restricted domain.

7. *Summary*

"Direct Search" techniques constitute an approach to a variety of numerical problems for which classical methods of solution have proved unfeasible. With their emphasis on the use of simple strategies rather than complex tactics, they are more suited to use on modern computers than classical methods and would appear to justify work done in seeking more precise definitions and results than we have been able to find. In addition to their practical importance in supplying a successful way of attacking hitherto "unsolvable" problems, we feel their greatest value may lie in stimulating new concepts for functional classification.

APPENDIX A

A Formal Definition of Direct Search

A formal definition of direct search is provided below. This definition was made to be general and to describe direct search in its most basic form. It is not difficult, however, to devise procedures that are not altogether covered by this definition.

There is a space Σ of points P. There is a comparative relation $\subset$ on the points of Σ satisfying the transitive relation $(P \subset Q, Q \subset R) \Rightarrow P \subset R$. There is an extremal point P^* of Σ with the property $P^* \subset P$ for any other P in Σ. The point P^* represents the solution to a problem, other points of Σ representing possible solutions.

A direct search procedure makes sequential comparisons, using the $\subset$ relation, to determine:

1. A set of "trial points" P_r. (Here and below, $r = 1, 2, \cdots, N$, where N is determined below.)
2. An initial "base point" B_0 and a set of base points B_r, where $B_r = P_s$ for some $s \leqq r$.
3. A set of integers, called the "states" of the procedure, including an initial state S_0, the state S_r being associated with the base point B_r.
4. A "stop rule" for terminating the procedure, that is, for determining N.
5. An "approximate solution", or approximation to P^*, which is B_N.

The "strategy" for the determination of these sets is defined by the following rules:

a. The initial base point B_0 and the initial state $S_0 (\neq 0)$ are arbitrary.
b. The remaining trial points are $P_r = h(B_{r-1}, S_{r-1})$, where h is a function onto Σ, and $S_{r-1} \neq 0$.

c. If $P_r \subset B_{r-1}$ then $B_r = P_r$ and $S_r = f(S_{r-1})$. Otherwise, $B_r = B_{r-1}$ and $S_r = g(S_{r-1})$.

d. When for the first time $S_i = 0$, the procedure stops; i.e., $N = i$.

Direct search is distinguished from other numerical procedures by having a finite number of states which, without loss of generality, can be indexed by a set of integers. It includes, as special cases, those techniques for which h does not depend on B_{r-1}, such as (a) techniques which depend on inspection of all points of an arbitrarily determined set, or (b) Monte Carlo techniques which depend on a random selection of points in Σ. It does not include methods which possess a continuum of states, such as those (e.g., Newton's method and methods of steepest ascent) which rely on the use of such tools as derivatives and power series approximations.

APPENDIX B

Pattern Search

An exact description of a pattern search routine, which has been applied, is given in the accompanying figures. A flow diagram for pattern search is given in Chart 1. The notation is self-explanatory. The sequence following the label ② is the basic iterative loop consisting of a pattern move followed by a set of exploratory moves. The sequence following the label ① is for an initial set of exploratory moves from a base point when a new pattern must be established. The sequence labeled ③ controls the reduction of step size and termination of the search.

The remaining charts (2–4) give details of the procedure. Explicitly the procedure is carried out by sequentially transforming a set of variables. These

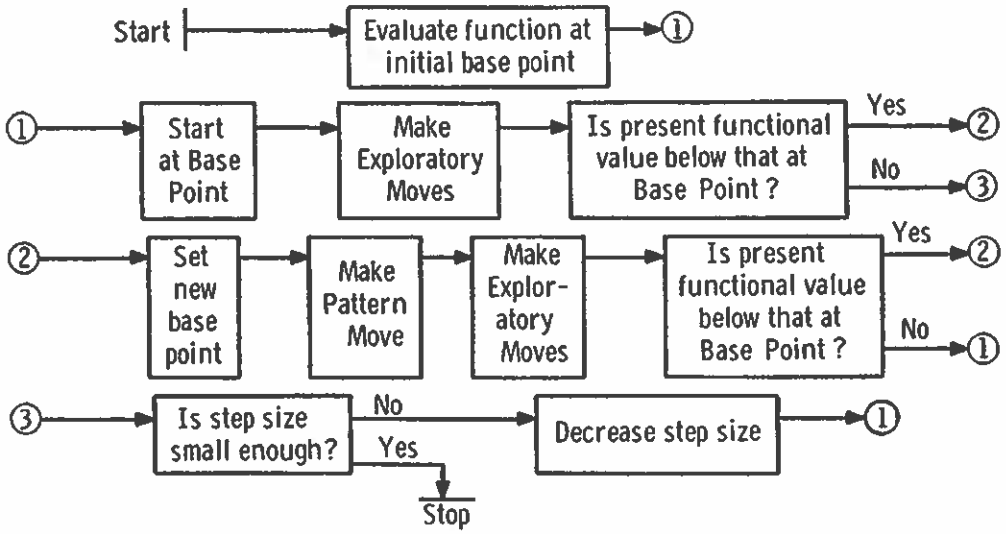


CHART 1. Descriptive flow diagram for pattern search

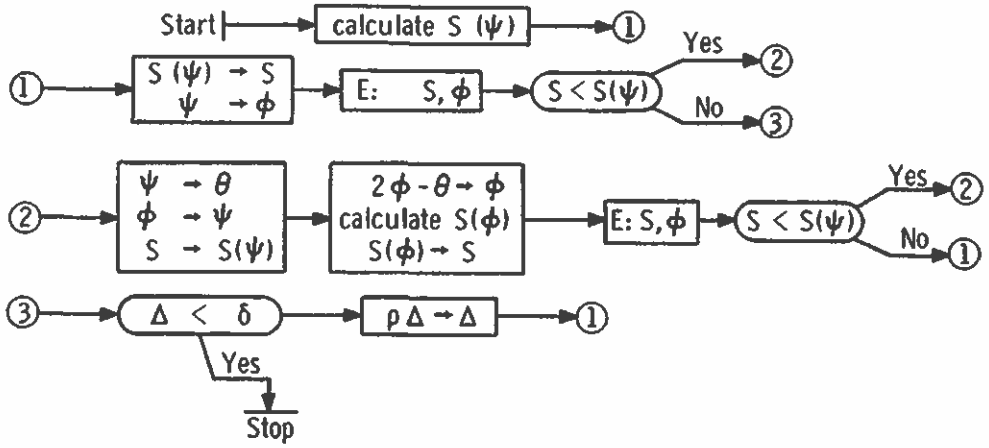


CHART 2. Detailed flow diagram for pattern search

For notation see Table IV.

Starting Conditions: Initially, the variables ψ and K as well as Δ, ρ, and δ are assigned values; and a routine to compute $S(\varphi)$ is supplied.

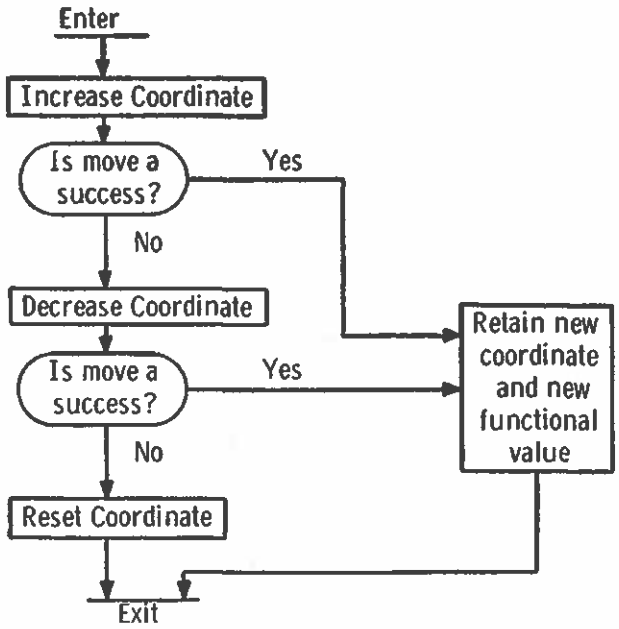


CHART 3. Descriptive flow diagram for exploratory moves (Program E).

The routine shown is carried out for each coordinate separately

variables and their value interpretations are given in Table III. In Chart 2, which for clarity has been drawn to parallel Chart 1, a detailed flow diagram in terms of the problem variables is exhibited.

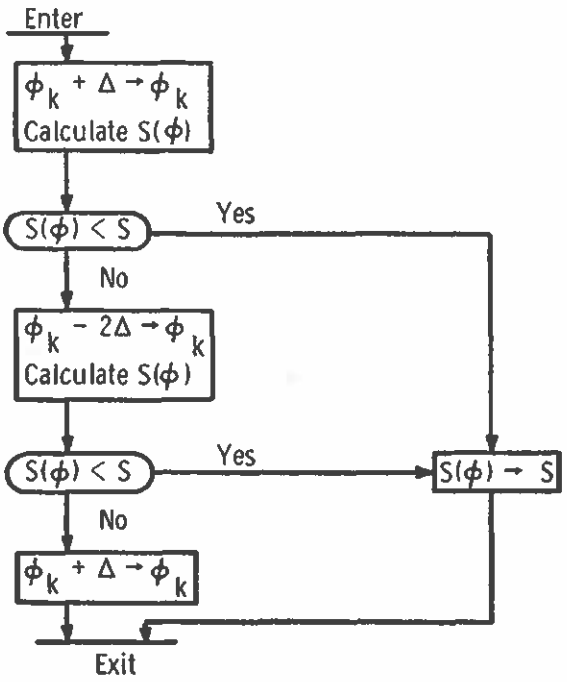
The notation is explained in Table IV. All branching conditions are given in terms of direct questions symbolized by a statement surrounded by an oval. The substitution transformation $x \rightarrow y$ is here defined to constitute a means of

CHART 4. Detailed flow diagram for exploratory moves

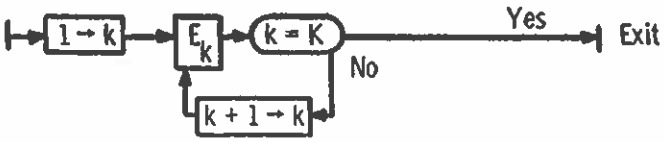
(a) The program E is 

(see note)

(b) The program E_k is:



NOTE: The loop implicit in (a) may be carried out explicitly (for $K \geqq 1$) by the routine:



or by any other suitable equivalent iteration scheme.

TABLE III

Variables and Their Value Interpretations for Charts 2 and 4

The variables ψ, θ, and φ are points in a K-dimensional space; the rest of the variables are unidimensional.

θ	the previous base point
ψ	the current base point
φ	the base point resulting from the current move
$S(\psi)$	the functional value at the base point
$S(\varphi)$	the functional value for this move
S	the functional value before this move (usually, the smallest value so far attained by the set of exploratory moves)
Δ	current step size
δ	"minimum" step size
ρ	reduction factor for step size, $\rho < 1$.
φ_k	one of the coordinate values for φ, $k = 1, 2, \cdots, K$.
K	number of coordinates for the points.

TABLE IV

Notation for Detailed Flow Diagrams

$x \rightarrow y$	means the value of the variable x is to become the new value of the variable y.
(p)	stands for the question "Is the statement p true?".
E: S, φ	indicates that a program E (See Charts 3 and 4) is to be carried out which will affect the values of the variables S and φ.
calculate $S(\varphi)$	indicates that the value of $S(\varphi)$ is to be calculated by an independent program

altering the value of the variable y. Programs specific to any particular electronic computer can be adapted from these flow diagrams.

APPENDIX C

Incremental Procedures

In calculating a sequence of functional values, time can usually be saved by simply calculating the changes in the successive values of the function. This is particularly true when the argument is changing in fixed increments, in which case multiplications can generally be replaced by additions and functional subroutines by either single multiplications or simple linear algorithms.

Consider, for example, the calculation of $y = a + be^{cx}$ for a single sequence of values of the arguments (a, b, c), using fixed simultaneous increments Δa, Δb, and Δc. A conventional computational scheme would consist of repetition of the sequence

$$\begin{aligned} a + \Delta a &\rightarrow a \\ b + \Delta b &\rightarrow b \\ c + \Delta c &\rightarrow c \\ cx &\rightarrow t \\ \exp(t) &\rightarrow \xi \\ a + b\xi &\rightarrow y, \end{aligned}$$

where exp indicates entrance into an exponential subroutine. A corresponding incremental scheme would be

$$\begin{aligned} a + \Delta a &\rightarrow a \\ b + \Delta b &\rightarrow b \\ \xi\eta &\rightarrow \xi \\ a + b\xi &\rightarrow y, \end{aligned}$$

where η is the fixed constant $e^{x\Delta c}$, the third line arising from

$$\xi + \Delta\xi = e^{(c+\Delta c)x} = e^{cx}e^{x\Delta c} = \xi\eta.$$

The big reduction in time, of course, comes from the replacement of the use of the exponential subroutine by a single multiplication.

The incremental scheme is clearly well adapted to repeated (unmodified) pattern moves and can be incorporated into the routine which computes the successive values of S. (Related schemes could be similarly applied to the S

computation itself, but these will not be detailed here.) A similar scheme can be applied to the exploratory moves—a separate scheme being used for each direction and each parameter—provided appropriate modifications are made, as described below, to "update" the pattern move scheme.

When an exploratory move is made in parameter a, the pattern move increment Δa must be modified by addition of the exploratory move increment δa. The value of δa is either α or $-\alpha$, where α is a constant. In carrying out the exploratory move, δa is used in lieu of Δa, and 0 in lieu of Δb and Δc, since only one parameter is being changed. Similarly, for an exploratory move in parameter b, δb (which is $\pm\beta$) must be added to Δb. Finally, for an exploratory move in parameter c, the multiplicative factor $\epsilon = e^{x\delta c}$ is used in lieu of η, and η is multiplied by ϵ; the value of ϵ is $e^{\gamma x}$ or $e^{-\gamma x}$, where γ is a constant. These schemes are summarized below:

EXPLORATORY MOVES:

(i) a parameter

$$\begin{aligned} a + \delta a &\to a \\ a + b\xi &\to y \quad \text{(or more simply } y + \delta a \to y\text{)} \\ \Delta a + \delta a &\to \Delta a \end{aligned}$$

(ii) b parameter

$$\begin{aligned} b + \delta b &\to b \\ a + b\xi &\to y \\ \Delta b + \delta b &\to \Delta b \end{aligned}$$

(iii) c parameter

$$\begin{aligned} \xi\epsilon &\to \xi \\ a + b\xi &\to y \\ \eta\epsilon &\to \eta \end{aligned}$$

PATTERN MOVES:

$$\begin{aligned} a + \Delta a &\to a \\ b + \Delta b &\to b \\ \xi\eta &\to \xi \\ a + b\xi &\to y \end{aligned}$$

REFERENCE

1. KIEFER, J. Sequential minimax search for a maximum *Proc. Amer. Math. Soc. 4* (1953), 502–506