SIAM REVIEW
Vol. 12, No. 1, January 1970

A SURVEY OF NUMERICAL METHODS FOR UNCONSTRAINED OPTIMIZATION*

M. J. D. POWELL†

Abstract. This paper is intended to introduce and relate the more successful numerical methods for calculating the greatest value of a given real function $F(x_1, x_2, \ldots, x_n)$. It also indicates some directions for research.

1. Introduction. We consider the problem of calculating the greatest value of a given real function $F(x_1, x_2, \cdots, x_n)$, where each variable x_i, $i = 1, 2, \cdots, n$, can take the value of any real number. We wish to carry out the calculation on a computer, so the objective function is usually defined by a subroutine that will evaluate $F(x_1, x_2, \cdots, x_n)$ for any choice of the vector of variables. Therefore the problem is solved by applying a systematic method to try many different values of the variables, the choice of variables being guided by the values of the objective function that have already been calculated. Some methods for adjusting the variables use strategies that depend on derivatives of $F(x_1, x_2, \cdots, x_n)$, in which case the computer subroutine that specifies the objective function has to be extended to calculate derivatives as well.

Because so many different algorithms have been suggested to solve the unconstrained optimization problem, this paper is intended to introduce the subject in a concise and uncomplicated way. Therefore we deliberately omit many details that are needed to write computer programs, because we try to emphasize the ideas that cause the success of the algorithms, rather than the techniques themselves. In particular we dwell on the ideas either that have proved to be basic to the subject, or that, in the author's opinion, are likely to yield methods that are more successful than the ones of today. Thus we indicate some areas of research, and here we must emphasize that at the present time the chances of finding substantial improvements to existing algorithms seem to be excellent.

Other papers that relate different methods of optimization have been written by Box [5], Dickinson [11], Fletcher [13], Leon [24], Powell [30] and Spang [34], and a more detailed introduction to the subject will soon be published, written by Box, Davies and Swann [6].

2. The objective function. In order to calculate the greatest value of the objective function, $F(x_1, x_2, \cdots, x_n)$, by a general algorithm, it is necessary for $F(x_1, x_2, \cdots, x_n)$ to satisfy some smoothness conditions. It is not possible to be specific, but it is appropriate to make some general remarks here, because we must indicate that we cannot expect any algorithm to be entirely successful.

The reason is that, because each component of the vector of variables $(x_1, x_2, \cdots, x_n)$ is free to take the value of any real number, we cannot begin to cover

* Received by the editors July 23, 1968. Presented by invitation at the Symposium on Optimization, sponsored by the Air Force Office of Scientific Research, at the 1968 National Meeting of Society for Industrial and Applied Mathematics, held in Toronto, Canada, June 11–14, 1968.

† Mathematics Branch, Theoretical Physics Division, Atomic Energy Research Establishment, Harwell, Berkshire, England.

the whole space of the variables. Further, even if from the structure of $F(x_1, x_2, \cdots, x_n)$ we know that the required maximum must lie in a finite region, say, $0 \leqq x_i \leqq 1$, $i = 1, 2, \cdots, n$, then often a covering is not feasible even for quite small values of n, because a square mesh of size h in the unit cube demands $(1 + h^{-1})^n$ points. Therefore, a realistic ambition is to develop a general algorithm whose failure will be infrequent when it is applied to problems of the real world.

Most algorithms economize on the number of function evaluations by searching near the point at which the largest value of $F(x_1, x_2, \cdots, x_n)$ has been calculated. In the case $n = 2$ it is easy to visualize this idea, for $z = F(x_1, x_2)$ describes a surface in three dimensions, and we require the point of the surface that is furthest above the plane $z = 0$. Also the analogy shows a disadvantage of the general method, which is that if the best calculated value of $F(x_1, x_2, \cdots, x_n)$ happens to be on a small local hill of the surface, then an algorithm is likely to find and to stop at the top of the small hill, even if there are higher hills. Because of the difficulty of searching right through the space of the variables, this deficiency seems to be inevitable in general optimization methods. Therefore, for guaranteed success, it is often necessary for $F(x_1, x_2, \cdots, x_n)$ to have only one maximum point, but unfortunately functions with more than one maximum are often encountered in practice.

If there are many maxima, the one that is found usually depends on an initial estimate of the solution, which has to be provided by the user of the algorithm. In this case different initial estimates can be tried if it is suspected that $F(x_1, x_2, \cdots, x_n)$ is awkward, but this device alone can only reduce, rather than eliminate, the possibility of failure.

For these reasons we judge an algorithm to be successful if it calculates a local maximum of $F(x_1, x_2, \cdots, x_n)$, from a given starting approximation to the required vector of variables. This statement and the remark that optimization can be visualized as a hill-climbing problem indicate the basic conditions that must be satisfied by the objective function. Also some algorithms require $F(x_1, x_2, \cdots, x_n)$ to be differentiable.

The lack of definiteness in stating the conditions on $F(x_1, x_2, \cdots, x_n)$ is deliberate, because we are describing the current state of optimization, and it happens that the current state is not a logical structure of theorems. Instead it has developed from an assortment of numerical methods which have been devised because real problems have had to be solved, and at present the actual success of the algorithms is far ahead of any theoretical predictions. Fortunately there seems to be an increase in the amount of study on theoretical questions, so the subject of unconstrained optimization should soon become more coherent.

3. Old methods. In §§ 3–7 we follow the history of optimization algorithms, and we mention various methods that have been in use for at least three years. However, in §§ 8–10 we consider some recent ideas, and also we suggest some future developments.

The celebrated steepest ascent method was described by Cauchy [8] in 1847. In the case $n = 2$, when the objective function is the surface $z = F(x_1, x_2)$, the method is obtained by asking the question, "Which uphill direction is steepest?" If the objective function is differentiable, and we take a very small step (δ_1, δ_2) from

the point (x_1, x_2), we obtain from first order terms in the Taylor series the result:

$$F(x_1 + \delta_1, x_2 + \delta_2) - F(x_1, x_2) \approx \delta_1 \frac{\partial F}{\partial x_1} + \delta_2 \frac{\partial F}{\partial x_2} \tag{1}$$
$$= \sqrt{\delta_1^2 + \delta_2^2} \sqrt{\left(\frac{\partial F}{\partial x_1}\right)^2 + \left(\frac{\partial F}{\partial x_2}\right)^2} \cos\theta,$$

where θ is the angle between the step (δ_1, δ_2) and the gradient vector at (x_1, x_2). Different directions of the step give different angles, and (1) suggests that the steepest direction occurs when $\cos\theta$ is equal to one. Therefore a step of the steepest ascent method from the point $(x_1, x_2, \cdots, x_n)$ is chosen to be in the direction $(g_1, g_2, \cdots, g_n)$, where g_i is defined by the equation:

$$g_i(x_1, x_2, \cdots, x_n) = \frac{\partial F(x_1, x_2, \cdots, x_n)}{\partial x_i}, \qquad i = 1, 2, \cdots, n. \tag{2}$$

The length of the step is often calculated to maximize the new value of the objective function, in which case an iteration replaces an estimate $(x_1, x_2, \cdots, x_n)$ of the position of the optimum by the estimate $(x_1 + \lambda^* g_1, x_2 + \lambda^* g_2, \cdots, x_n + \lambda^* g_n)$, where λ^* is the value of λ that maximizes the function of one variable:

$$\phi(\lambda) = F(x_1 + \lambda g_1, x_2 + \lambda g_2, \cdots, x_n + \lambda g_n). \tag{3}$$

Many other optimization algorithms are like the steepest ascent method, for they calculate a direction at an estimate $(x_1, x_2, \cdots, x_n)$, and they change the estimate by a multiple of the direction, the multiplier being chosen to maximize the new value of $F(x_1, x_2, \cdots, x_n)$. Therefore the problem of calculating the maximum value of a function of one variable, like expression (3), often needs to be solved. We prefer not to describe the methods of solution here (techniques are given in references [6], [14], [29] and [34]), because most of the details do not contribute to the understanding of many variable problems. However, there is one detail that should be mentioned, which is that it is usually inefficient to seek high accuracy, when the one-dimensional problem is derived from a poor estimate $(x_1, x_2, \cdots, x_n)$ of the solution of the n-dimensional optimization. Box [5] makes some interesting remarks on this question.

The steepest ascent algorithm is one of the few optimization methods that is supported by convergence theorems [20]. If $F(x_1, x_2, \cdots, x_n)$ is bounded above, and has a uniformly continuous first derivative, and if each function $\phi(\lambda)$ (see (3)) has a maximum, then the successive gradient vectors $(g_1, g_2, \cdots, g_n)$ that are generated by the iterations of the algorithm will converge to zero. Therefore any limit point of the sequence of estimates $(x_1, x_2, \cdots, x_n)$ must be a stationary point of $F(x_1, x_2, \cdots, x_n)$. Further, if both the initial estimate $(x_1, x_2, \cdots, x_n)$ and the objective function are such that all the estimates $(x_1, x_2, \cdots, x_n)$ are in a compact space, and also if $F(x_1, x_2, \cdots, x_n)$ has separate stationary points and is not constant along the straight line joining any two of its stationary points (this last condition rules out the possibility that the iterations oscillate between one stationary point and another), then the steepest ascent algorithm will converge to a distinct stationary point of the objective function. The point need not be a maximum, but fortunately it usually happens that convergence to a stationary point that is not a maximum is unstable.

The rate of convergence has been analyzed by Akaike [1], and has been experienced by very many computer users. It is often intolerably slow, so the algorithms described later in this paper are usually more satisfactory. Indeed the poor performance of the steepest ascent method was a strong motivation in the development of recent techniques.

Often it happens that first derivatives of the objective function are not available, either because they do not exist, or, more usually, because one prefers not to calculate them. In this case a classical method of optimization is to adjust each variable separately. Specifically, the initial estimate $(x_1, x_2, \cdots, x_n)$ is altered to $(x_1 + \lambda_1, x_2, \cdots, x_n)$, this estimate is altered to $(x_1 + \lambda_1, x_2 + \lambda_2, x_3, \cdots, x_n)$, and so on until the nth stage of the process replaces $(x_1 + \lambda_1, x_2 + \lambda_2, \cdots, x_{n-1} + \lambda_{n-1}, x_n)$ by $(x_1 + \lambda_1, x_2 + \lambda_2, \cdots, x_n + \lambda_n)$. These n stages are repeated, usually until the corrections $(\lambda_1, \lambda_2, \cdots, \lambda_n)$ become very small. One way of obtaining the values of the numbers λ_i is to use a one-dimensional maximization algorithm to make each new value of the objective function as large as possible.

Because this process can increase $F(x_1, x_2, \cdots, x_n)$ unless all the gradient components g_i, $i = 1, 2, \cdots, n$, are zero, the convergence properties are like those of the steepest ascent algorithm. In particular the rate of convergence is usually too slow to be useful. However, sometimes the structure of $F(x_1, x_2, \cdots, x_n)$ is such that the successive evaluations of the objective function can be speeded up considerably, because the algorithm changes only one of the n variables at each stage.

4. Newton-Raphson. As its name suggests, the "Newton-Raphson" algorithm is also an "old method", but we describe it in a new section because it is still the best way of solving certain problems. Its special feature is that it takes account of second derivatives of $F(x_1, x_2, \cdots, x_n)$. Second derivative terms are absolutely basic to unconstrained maximization, because a differentiable function cannot have a distinct maximum point unless it curves. At this stage our ideas depart from the concepts of linear programming.

The Newton-Raphson method estimates the position of the maximum of $F(x_1, x_2, \cdots, x_n)$ from second and lower order terms in the Taylor series. Specifically it uses the approximation:

$$\begin{aligned} &F(x_1 + \delta_1, x_2 + \delta_2, \cdots, x_n + \delta_n) \\ &\quad \approx F(x_1, x_2, \cdots, x_n) + \sum_{i=1}^{n} \delta_i g_i + \tfrac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \delta_i G_{ij} \delta_j, \end{aligned} \tag{4}$$

where g_i and G_{ij} are defined by (2) and by the equation:

$$G_{ij} = \frac{\partial^2 F(x_1, x_2, \cdots, x_n)}{\partial x_i \partial x_j}. \tag{5}$$

At the maximum of a differentiable function all first derivatives are equal to zero, so, if (4) is exact, the point $(x_1 + \delta_1, x_2 + \delta_2, \cdots, x_n + \delta_n)$ is the required optimum only if the equations

$$g_i + \sum_{j=1}^{n} G_{ij} \delta_j = 0, \qquad i = 1, 2, \cdots, n, \tag{6}$$

are satisfied. But these equations are linear in the components δ_i, so it is usually straightforward to calculate their solution:

$$\delta_i = -\sum_{j=1}^{n} (G^{-1})_{ij} g_j, \qquad i = 1, 2, \cdots, n. \tag{7}$$

Because of these remarks the Newton-Raphson method changes an estimate $(x_1, x_2, \cdots, x_n)$ of the required vector of variables to the estimate $(x_1 + \delta_1, x_2 + \delta_2, \cdots, x_n + \delta_n)$, where the numbers δ_i are defined by (7), the derivatives G_{ij} and g_j being calculated at the point $(x_1, x_2, \cdots, x_n)$. This technique is applied iteratively, by changing each new estimate in the way described, and it is hoped that the resultant sequence of vectors $(x_1, x_2, \cdots, x_n)$ will converge to the values of the variables that maximize the objective function.

Because the Newton-Raphson method is exact if the approximation (4) holds, it may be proved that, if the second derivative matrix G is negative definite at the solution, then the iterations have quadratic convergence, provided that the initial estimate $(x_1, x_2, \cdots, x_n)$ is sufficiently close to the optimal values of the variables. Therefore the procedure can be very efficient, but, if the initial estimate is poor, the method often fails to converge. Even when it does converge it may reach a point which is not a maximum, because the derivation of an iteration depends only on the fact that the first derivatives of $F(x_1, x_2, \cdots, x_n)$ are zero at a maximum, and the first derivatives are zero at all stationary points of the objective function.

To avoid convergence to a stationary point that is not a maximum, and to ensure that an iteration does not worsen the value of the objective function, a valuable well-known strategy is to use the correction $(\delta_1, \delta_2, \cdots, \delta_n)$, defined by (7), as a direction of search in the space of the variables. (Moreover, (6) can define a direction of search even when G is singular.) Specifically, we treat the function of one variable:

$$\phi(\lambda) = F(x_1 + \lambda\delta_1, x_2 + \lambda\delta_2, \cdots, x_n + \lambda\delta_n) \tag{8}$$

in the same way as we treated the function (3). Thus we obtain an algorithm that has both second order convergence, and the ability to converge from very poor starting approximations to the required solution.

This "extended Newton-Raphson" algorithm has been very successful in practice, but there has been little work on identifying general conditions for guaranteed convergence to a local maximum. Powell [30] gives an example to show that the method can fail at a point where the gradient of $F(x_1, x_2, \cdots, x_n)$ is nonzero, and G is nonsingular, so probably any theoretical conditions would exclude many of the successful solutions to real problems that have been obtained already.

It is interesting and instructive to note that the extended Newton-Raphson algorithm is identical to the steepest ascent method, if the second derivative matrix G is equal to minus the unit matrix. It is instructive because, if $-G$ is positive definite (this condition is usually obtained at the solution), and we make the change of variables:

$$y_i = \sum_{j=1}^{n} (-G^{1/2})_{ij} x_j, \tag{9}$$

then the second derivative matrix of the right-hand side of (4) becomes equal to minus the unit matrix. For this reason each iteration of the steepest ascent algorithm can be made as successful as a Newton-Raphson iteration by applying a suitable linear transformation to the variables, but the best linear transformation depends on $(x_1, x_2, \cdots, x_n)$. This idea is developed in our discussion of Davidon's method (see § 7).

To many computer users, the most serious disadvantage of the Newton-Raphson algorithm is that it requires second derivatives of the objective function. Therefore techniques have been developed to achieve fast ultimate convergence, by taking the curvature of $F(x_1, x_2, \cdots, x_n)$ into account, without the explicit evaluation of second derivatives. Many of these methods are described later in this paper.

5. Extensions of the old methods. As soon as automatic computers were used to apply the steepest ascent and Newton-Raphson algorithms, it became worthwhile to overcome the limitations of the techniques. In particular attempts were made [4], [12], [17] to find modifications to improve the slow convergence of the steepest ascent method, and to make the Newton-Raphson iteration more reliable.

Booth [4] remarks that it is sometimes worthwhile to modify the steepest ascent method so that it takes shorter steps. Specifically he suggests applying the classical method only every fifth iteration, while on the remaining iterations he replaces the current estimate $(x_1, x_2, \cdots, x_n)$ of the solution by the estimate $(x_1 + 0.9\,\lambda^* g_1, x_2 + 0.9\,\lambda^* g_2, \cdots, x_n + 0.9\,\lambda^* g_n)$, where λ^* is calculated to maximize (3). A good reason for taking these shorter steps is obtained by asking the question: given that we shall apply just two iterations of the steepest ascent method, except that we are free to choose the step-length of the first iteration, what choice of this step-length will cause the final value of $F(x_1, x_2, \cdots, x_n)$ to be greatest? If $F(x_1, x_2, \cdots, x_n)$ is a negative definite quadratic function, it is straightforward to analyze this question, and it happens that, in the usual case when the gradient vector does not point directly at the solution, it is best to choose the initial step to be less than the ordinary step. Presumably, because of the dependence of the steepest ascent method on linear transformations of the variables (see the remarks supporting (9)), the same device can be used to advantage when the direction of search is obtained by multiplying the gradient vector by a constant positive definite matrix. A similar technique is described by Fadeev and Fadeeva [12], and they give an example to show that the shorter steps can provide a faster rate of convergence.

The modification suggested by Forsythe and Motzkin [17] has a more satisfactory mathematical basis. They noticed that the steepest ascent method usually converges in such a way that the asymptotic behavior of the successive estimates $(x_1, x_2, \cdots, x_n)$ is that they alternate between just two lines (in the n-dimensional space of the variables) which meet at the required solution. Some theorems on this behavior have been given by Akaike [1]. The directions of these lines tend to be along the vectors $(x_1^{(k-2)} - x_1^{(k)}, x_2^{(k-2)} - x_2^{(k)}, \cdots, x_n^{(k-2)} - x_n^{(k)})$, where the superscripts are iteration numbers, so the modification introduces occasional searches along these directions. Specifically the estimate $(x_1^{(k+1)}, x_2^{(k+1)}, \cdots, x_n^{(k+1)})$ is calculated usually by searching from $(x_1^{(k)}, x_2^{(k)}, \cdots, x_n^{(k)})$ along the direction

$(g_1^{(k)}, g_2^{(k)}, \cdots, g_n^{(k)})$ for the maximum of $F(x_1, x_2, \cdots, x_n)$, but sometimes the direction of search is $(x_1^{(k-2)} - x_1^{(k)}, x_2^{(k-2)} - x_2^{(k)}, \cdots, x_n^{(k-2)} - x_n^{(k)})$.

A third extension to the steepest ascent algorithm is the "s-step gradient method" [12]. It has been used when the objective function is quadratic, and in this case each iteration is related to s, $1 < s < n$, consecutive iterations of the method of steepest ascents. Specifically an iteration, instead of maximizing the one-dimensional function (3) to provide the best point in the direction $(g_1, g_2, \cdots, g_n)$, calculates the value of $(\lambda_1, \lambda_2, \cdots, \lambda_s)$ that maximizes the function of s variables:

$$\phi(\lambda_1, \lambda_2, \cdots, \lambda_s) = F\left\{(x_1, x_2, \cdots, x_n) + \sum_{t=1}^{s} \lambda_t(g_{t1}, g_{t2}, \cdots, g_{tn})\right\}, \tag{10}$$

where the directions $(g_{t1}, g_{t2}, \cdots, g_{tn})$, $t = 1, 2, \cdots, s$, are the gradients that would be used by s iterations of the steepest ascent method. Using the notation $(\lambda_1^*, \lambda_2^*, \cdots, \lambda_s^*)$ for the calculated vector of parameters, an iteration replaces the estimate $(x_1, x_2, \cdots, x_n)$ by the estimate:

$$(x_1, x_2, \cdots, x_n) + \sum_{t=1}^{s} \lambda_t^*(g_{t1}, g_{t2}, \cdots, g_{tn}). \tag{11}$$

Forsythe [16] has indicated that the asymptotic behavior of the s-step gradient method is like that of the method of steepest ascents, so we prefer the more recent algorithms that are described in § 7.

An extension to make the Newton-Raphson method more reliable was suggested by Levenberg [25] in 1944, but he applies the idea to the problem of solving overdetermined algebraic equations. A description that is specific to unconstrained optimization is given by Goldfeld, Quandt and Trotter [18], [19]. The extension introduces a nonnegative parameter α that interpolates between the steepest ascent iteration and the Newton-Raphson formula (7), the actual iteration being the replacement of an estimate $(x_1, x_2, \cdots, x_n)$ by the estimate $(x_1 + \delta_1, x_2 + \delta_2, \cdots, x_n + \delta_n)$, where the correction is defined by the equation:

$$\delta_i = \sum_{j=1}^{n} ([\alpha I - G]^{-1})_{ij} g_j, \qquad i = 1, 2, \cdots, n, \tag{12}$$

I being the unit matrix. Note that in the case $\alpha = 0$, equation (12) reduces to Newton-Raphson, while if α becomes very large the correction to $(x_1, x_2, \cdots, x_n)$ tends to have the direction of the gradient $(g_1, g_2, \cdots, g_n)$. Moreover, because large values of α cause the length of the correction to be small, there is no need to include a linear search procedure in an iteration. This seems to be one of the most promising ideas for improving existing algorithms, so we shall discuss it further in § 9.

6. Direct search methods. The introduction of automatic computers, as well as stimulating the extensions to classical methods that have just been described, also led to many optimization algorithms that do not require the calculation of derivatives. Most of these methods were guided by extensive numerical experience, and by thinking of the problem as that of climbing a hill, so they developed in an intuitive way. Some very useful techniques resulted, and we mention a few of them

now. In the next section other techniques are given that also do not require derivatives to be computed, but they are described separately because, unlike the methods of this section, they originated from an intention to take account of low order terms in a Taylor series expansion of the objective function.

To begin the development of the direct search algorithms, we consider the classical method, described in § 3, of revising the variables of the objective function one at a time. We continue to define an iteration as the process which adjusts each variable once only, and we consider the estimates $(x_1^{(k)}, x_2^{(k)}, \cdots, x_n^{(k)})$, $k = 1, 2, \cdots$, that are obtained by the successive iterations. In hill-climbing terminology, it often happens that this process generates estimates that climb up to the top of a ridge, and then they follow the ridge to the solution, and in fact most of the computer time is usually spent in following the ridge. Rosenbrock noticed that the points on a ridge are often nearly collinear, and he used this remark to provide a successful algorithm [31].

The basis of Rosenbrock's method is that it tries to identify the direction of the ridge in order to use it as a search direction. Initially the variables are changed one at a time, as in the classical method, so on the first iteration the initial estimate $(x_1, x_2, \cdots, x_n)$ is changed to $\{(x_1, x_2, \cdots, x_n) + \lambda_1 \mathbf{d}_1\}$, this estimate is changed to $\{(x_1, x_2, \cdots, x_n) + \lambda_1 \mathbf{d}_1 + \lambda_2 \mathbf{d}_2\}$, and so on, until the complete iteration replaces the initial guess of the solution by the estimate:

$$(x_1, x_2, \cdots, x_n) + \sum_{t=1}^{n} \lambda_t \mathbf{d}_t, \tag{13}$$

where, for $t = 1, 2, \cdots, n$, $\mathbf{d}_t$ is the tth coordinate direction in the space of the variables. However, before starting a new iteration, the set of n search directions is changed, and the first search direction is replaced by the vector:

$$\mathbf{d}_1^* = \sum_{t=1}^{n} \lambda_t \mathbf{d}_t, \tag{14}$$

which is the change to the estimate of the solution that has just been calculated. The remaining new search directions are obtained by an orthogonalization process, and then the iterative process is repeated. Thus, if a straight ridge is followed to the solution, the first search direction becomes aligned along the ridge, and so the rate of convergence becomes faster. A further development of this idea is due to Davies, Swann and Campey [37], and some numerical examples are given by Box [5] and Fletcher [13].

Another useful direct search algorithm is described by Hooke and Jeeves [23]. It also seeks to identify the direction of ridges, but it has an important new feature that causes it usually to be even more successful than Rosenbrock's method. This feature comes from the remark that if a long straight step is taken along a curved ridge, then probably the end of the step will fall below the top of the ridge, but this does not matter because it is easy to identify a direction back to the top. Therefore, unlike most other methods, the algorithm of Hooke and Jeeves will sometimes change an estimate $(x_1, x_2, \cdots, x_n)$ of the solution, even though the change causes the value of the objective function to decrease.

An iteration of the Hooke and Jeeves algorithm is in two parts, which are called "pattern move" and "exploratory move". The pattern move is applied

first, and it changes the current estimate $(x_1^{(k)}, x_2^{(k)}, \cdots, x_n^{(k)})$ of the position of the solution by the total change made in the last iteration (except that on the first iteration there is no pattern move); let the resultant point be $(y_1, y_2, \cdots, y_n)$. From this point an exploratory move is made, which is really a fine adjustment of the values of the variables. Specifically small steps are taken along each of the coordinate directions in order to increase the objective function; let the resultant point be $(z_1, z_2, \cdots, z_n)$. If $F(z_1, z_2, \cdots, z_n)$ is greater than $F(x_1^{(k)}, x_2^{(k)}, \cdots, x_n^{(k)})$, then $(z_1, z_2, \cdots, z_n)$ becomes the starting approximation for the next iteration, but otherwise the iteration is treated as a failure, and instead an exploratory move is made from the point $(x_1^{(k)}, x_2^{(k)}, \cdots, x_n^{(k)})$.

It should be clear that the pattern moves can take long steps along ridges, while the exploratory phase can climb back to the crest of a ridge. The fact that $F(y_1, y_2, \cdots, y_n)$ is permitted to be less than $F(x_1, x_2, \cdots, x_n)$ is the special feature that causes Hooke and Jeeves method to be particularly suitable for optimizing objective functions that have curved ridges.

The method due to Baer [2] also includes the ideas that are important to Hooke and Jeeves algorithm. The main difference is that Baer uses first derivatives of the objective function, so his exploratory move is a search in the steepest ascent direction. Leon [24] relates these direct search methods to other algorithms.

Another valuable addition to the collection of optimization algorithms was made by Spendley, Hext and Himsworth [35]. It is called the "simplex method", and is quite different from the linear programming technique having the same name. A special property of the simplex method for unconstrained optimization is that its progress depends just on whether certain values of $F(x_1, x_2, \cdots, x_n)$ are greater than or less than other values of the objective function, so it is insensitive to inaccurate function values. Therefore it is a suitable technique when substantial random errors are present in the method for obtaining values of $F(x_1, x_2, \cdots, x_n)$. For instance, if the output of an industrial process depends on time and on certain adjustable input parameters, and if it is required to control these parameters so that the output remains near its optimal level, then the simplex method can be used to regulate the input variables, according to measurements of the objective function.

Its name is derived from the fact that it uses simplexes, a simplex in n-dimensional space being a solid having plane faces and $n + 1$ vertices. For example, triangles and tetrahedra are simplexes in two and three dimensions. The algorithm starts with a given simplex in the space of the variables, and then an iterative procedure is applied to move the given simplex to a position that is near to the optimal point in the space of the variables. Specifically each iteration moves the given simplex by reflecting it in one of its faces, so the position of only one vertex is changed by an iteration. The vertex that is moved is usually the one at which the value of the objective function is least, but there are exceptions to this rule to prevent premature cycling. Once the simplex is close to the optimal point, it can be contracted, in order to determine the required vector of variables more precisely.

Nelder and Mead [27] have extended the simplex method to allow reflection, or expansion, or contraction of the simplex on every iteration. Thus they obtain a procedure that appears to compare favorably with other good algorithms when the objective function is exact and differentiable. However, experiments by Box [5]

on objective functions having five and more variables indicate that Nelder and Mead's method is a good algorithm for optimizing an exact function only if the number of variables is small.

7. Conjugate direction methods. We remarked in § 4 that if the first few terms of a Taylor series expansion of the objective function are to be used to predict the solution of the unconstrained optimization problem, then second derivatives must be taken into account. Thus we obtained the Newton-Raphson algorithm, which requires the explicit evaluation of first and second derivatives of the objective function, and we stated that this method has second order convergence. Because this fast convergence is useful if very accurate answers are required, and because the task of calculating second derivatives can be laborious, various algorithms have been proposed that try to take account of second derivative terms, using only values and first derivatives of $F(x_1, x_2, \cdots, x_n)$. Some are described in this section, and we shall find that they use strategies that would yield the exact answer if $F(x_1, x_2, \cdots, x_n)$ were a quadratic function. This property alone does not guarantee fast convergence when the higher derivatives of $F(x_1, x_2, \cdots, x_n)$ are nonzero, but in practice the algorithms are extremely successful, although theoretical reasons for the success have not yet been found. Moreover, numerical examples show that often the methods of this section are the best available, even when the user's initial estimate of the vector of variables $(x_1, x_2, \cdots, x_n)$ is very far from the position of the optimum.

The methods calculate the exact maximum of a quadratic function by using "conjugate directions". We define the directions $\mathbf{p}$ and $\mathbf{q}$ in the space of the variables to be conjugate with respect to the negative definite (the condition of negative definiteness is needed to provide a solution to the optimization problem) quadratic objective function:

$$\Phi(x_1, x_2, \cdots, x_n) = c + \sum_{i=1}^{n} a_i x_i + \tfrac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} G_{ij} x_i x_j \tag{15}$$

if they are both nonzero, and if they satisfy the equation

$$\sum_{i=1}^{n} \sum_{j=1}^{n} p_i G_{ij} q_j = 0. \tag{16}$$

The reason they are useful is that if we search in the direction $\mathbf{p}$, and find the point $(x_1^{(k)}, x_2^{(k)}, \cdots, x_n^{(k)})$ that maximizes $\Phi(x_1^{(k)}, x_2^{(k)}, \cdots, x_n^{(k)})$, and then we search from $(x_1^{(k)}, x_2^{(k)}, \cdots, x_n^{(k)})$ in the conjugate direction $\mathbf{q}$ to reach the new estimate $(x_1^{(k+1)}, x_2^{(k+1)}, \cdots, x_n^{(k+1)})$, then the new value of the objective function cannot be increased by immediately searching again in the direction $\mathbf{p}$. (This statement is proved by straightforward algebra based on the observation that $(x_1, x_2, \cdots, x_n)$ is a best point in the direction $\mathbf{p}$ if and only if the gradient of the objective function at $(x_1, x_2, \cdots, x_n)$ is orthogonal to $\mathbf{p}$.) Using this remark, if we are given n mutually conjugate directions $\mathbf{d}_1, \mathbf{d}_2, \cdots, \mathbf{d}_n$, then we can calculate the exact maximum of the quadratic function $\Phi(x_1, x_2, \cdots, x_n)$ by the following process.

We start with an arbitrary estimate $(x_1^{(1)}, x_2^{(1)}, \cdots, x_n^{(1)})$ of the required variables, and then we search along the n conjugate directions. Specifically, for $k = 1, 2, \cdots, n$, the kth search replaces the estimate $(x_1^{(k)}, x_2^{(k)}, \cdots, x_n^{(k)})$ by the

estimate:

$$(x_1^{(k+1)}, x_2^{(k+1)}, \cdots, x_n^{(k+1)}) = (x_1^{(k)}, x_2^{(k)}, \cdots, x_n^{(k)}) + \lambda_k \mathbf{d}_k, \tag{17}$$

where λ_k is calculated to maximize the value of $\Phi(x_1^{(k+1)}, x_2^{(k+1)}, \cdots, x_n^{(k+1)})$. At the final point $(x_1^{(n+1)}, x_2^{(n+1)}, \cdots, x_n^{(n+1)})$ the gradient of the objective function must be orthogonal to the n directions $\mathbf{d}_1, \mathbf{d}_2, \cdots, \mathbf{d}_n$, but (16) can be used to show that mutually conjugate directions are independent. Therefore the n linear searches find the position of the optimum.

To use these ideas to optimize a given quadratic function is straightforward if the second derivative matrix is given, because one just has to calculate a set of n mutually conjugate directions to satisfy conditions like (16). In this case it is usual to choose $\mathbf{d}_1$ first, and then $\mathbf{d}_2$, and so on, in which case, if we suppose that the length of a direction is unimportant, there are $n - k$ degrees of freedom in the choice of $\mathbf{d}_k$. These degrees of freedom give many different methods, which are related to the various conjugate gradient algorithms for solving linear systems [22]. However, usually the objective function is not quadratic, so the concept of conjugacy is not defined. Therefore we must look for ways of simulating the condition (16) that will yield exact conjugacy in the quadratic case.

Zoutendijk [40] suggests one method that has not been used much, but it could prove to be a very valuable idea. He remarks that if the search direction $\mathbf{p}$ is used to move from the estimate $(x_1^{(k)}, x_2^{(k)}, \cdots, x_n^{(k)})$ to the estimate $(x_1^{(k+1)}, x_2^{(k+1)}, \cdots, x_n^{(k+1)})$, then, in the case that the objective function is quadratic, the condition (16) is the same as the equation

$$\sum_{i=1}^{n} (g_i^{(k+1)} - g_i^{(k)}) q_i = 0, \tag{18}$$

where $g_i^{(k)}$ is the ith component of the gradient of the objective function at $(x_1^{(k)}, x_2^{(k)}, \cdots, x_n^{(k)})$. Equation (18) is valuable because it contains no explicit second derivative terms, so we have a means of obtaining conjugacy when only values and first derivatives of $F(x_1, x_2, \cdots, x_n)$ are available. For instance we could start with an arbitrary search direction $\mathbf{d}_1$, and then, for $k = 2, 3, \cdots, n$, we could calculate the search direction $\mathbf{d}_k$ to be orthogonal to the changes in the gradient vector that were caused by the moves in the directions $\mathbf{d}_1, \mathbf{d}_2, \cdots, \mathbf{d}_{k-1}$.

The method just described simplifies in a remarkable and useful way if the first search direction is the direction of steepest ascent, and the later search directions $\mathbf{d}_k$, $k = 2, 3, \cdots$, are chosen to be the gradient $(g_1^{(k)}, g_2^{(k)}, \cdots, g_n^{(k)})$ plus the appropriate linear combination of the previous directions $\mathbf{d}_1, \mathbf{d}_2, \cdots, \mathbf{d}_{k-1}$, that provides conjugacy when $F(x_1, x_2, \cdots, x_n)$ is quadratic. In this case the required mutual conjugacy of the search directions can be obtained simply by choosing $\mathbf{d}_k$ to be of the form:

$$\mathbf{d}_k = (g_1^{(k)}, g_2^{(k)}, \cdots, g_n^{(k)}) + \beta_{k-1} \mathbf{d}_{k-1}, \tag{19}$$

where β_{k-1} is a real number. Moreover, β_{k-1} can be calculated from the formula

$$\beta_{k-1} = \sum_{i=1}^{n} [g_i^{(k)}]^2 \Big/ \sum_{i=1}^{n} [g_i^{(k-1)}]^2, \tag{20}$$

because it happens that, in the quadratic case, the process causes the successive gradients $(g_k^{(k)}, g_2^{(k)}, \cdots, g_n^{(k)})$, $k = 1, 2, \cdots$, to be mutually orthogonal. This method is the Fletcher–Reeves [15] algorithm, and it has been used successfully on many problems. The iteration is so elegant that it is straightforward to apply it any number of times when the objective function is not quadratic. However, its success on a quadratic function depends on the first iteration being a steepest ascent step, and after n iterations the identity of the first step must be lost. Therefore, to obtain a fast rate of convergence, Fletcher and Reeves recommend that the algorithm should be restarted with a steepest ascent step after every $n + 1$ iterations. An important advantage of the method, which is not obtained by other conjugate direction algorithms, is that it does not require storage space for any $n \times n$ matrices.

Conjugate directions have also been used to construct a successful algorithm [29] that does not require the explicit evaluation of any derivatives. When derivatives are not available conjugate directions are generated by using the following fact: if two separate points, say, $(y_1, y_2, \cdots, y_n)$ and $(z_1, z_2, \cdots, z_n)$, in the space of the variables are both obtained by searching along a direction, say $\mathbf{p}$, for the maximum value of the objective function, and if the objective function is quadratic, then the direction

$$\mathbf{q} = (y_1, y_2, \cdots, y_n) - (z_1, z_2, \cdots, z_n) \tag{21}$$

is conjugate to $\mathbf{p}$. (This property is also used in [28], [32] and [33].) An iteration of Powell's algorithm requires an estimate $(x_1^{(k)}, x_2^{(k)}, \cdots, x_n^{(k)})$ of the position of the optimum, and n independent search directions, say $\mathbf{d}_1^{(k)}, \mathbf{d}_2^{(k)}, \cdots, \mathbf{d}_n^{(k)}$, which initially are chosen to be the coordinate directions. Each iteration can change both the estimate and the search directions, and the algorithm is designed so that, if it is applied to a negative definite quadratic function, then usually n iterations will find the optimum, because the search directions become mutually conjugate. An important feature of an iteration is that special measures are taken to prevent the n search directions from becoming linearly dependent, but we omit these measures from our description of an iteration in order to make the main idea more easy to understand.

The simplified iteration begins by searching along each of the directions in turn, and thus the estimate $(x_1^{(k)}, x_2^{(k)}, \cdots, x_n^{(k)})$ is changed by the amount

$$\boldsymbol{\delta} = \sum_{i=1}^{n} \lambda_i \mathbf{d}_i^{(k)}, \tag{22}$$

where, for $i = 1, 2, \cdots, n$, the multiplier λ_i is calculated to maximize the intermediate value of the objective function

$$F\left\{(x_1^{(k)}, x_2^{(k)}, \cdots, x_n^{(k)}) + \sum_{j=1}^{i} \lambda_i \mathbf{d}_j^{(k)}\right\}. \tag{23}$$

Then the displacement $\boldsymbol{\delta}$, defined by (22), is also used as a search direction, and the resultant point, say $(x_1^{(k+1)}, x_2^{(k+1)}, \cdots, x_n^{(k+1)})$, is the starting estimate for the

next iteration Finally the search directions are replaced by the set

$$\begin{aligned} \mathbf{d}_i^{(k+1)} &= \mathbf{d}_{i+1}^{(k)}, && i = 1, 2, \cdots, n-1, \\ \mathbf{d}_n^{(k+1)} &= \boldsymbol{\delta}. \end{aligned} \tag{24}$$

The result of applying this process to a quadratic function is that usually the new directions generated by the kth iteration, namely $\mathbf{d}_1^{(k+1)}, \mathbf{d}_2^{(k+1)}, \cdots, \mathbf{d}_n^{(k+1)}$, have the property that the last k of them are mutually conjugate. This remark is proved by induction [29], unless linear dependence occurs [39], when the special measures of the complete algorithm become essential. Some numerical comparisons of this method with direct search techniques have been made by Box [5] and by Fletcher [13].

The last conjugate direction algorithm that we shall consider is probably the most successful of all. It is due to Davidon [9], but, because the original description has limited circulation, the paper by Fletcher and Powell [14] is recommended. Originally the algorithm was called a "variable metric" method, and we shall describe it from this point of view, because it is rather difficult to visualize a description that is based on conjugate directions.

To introduce the description, we note that the search direction $(\delta_1, \delta_2, \cdots, \delta_n)$ of the steepest ascent iteration at the point $(x_1, x_2, \cdots, x_n)$ is given by the formula

$$\delta_i = \sum_{j=1}^{n} I_{ij} g_j, \qquad i = 1, 2, \cdots, n, \tag{25}$$

where I is the unit matrix, and g_j is the jth component of the gradient of the objective function (see (2)). Also we note that, if in place of the unit matrix in (25) we substitute any positive definite matrix, then we obtain another search direction that has the property that its angle with the gradient vector is less than ninety degrees, so, provided that the gradient is nonzero, a search along this new direction will increase the objective function. In particular, in view of the remarks made at the end of § 4, some choice of this positive definite matrix will provide the fast convergence of the Newton-Raphson iteration. Therefore the kth iteration of Davidon's method changes the estimate $(x_1^{(k)}, x_2^{(k)}, \cdots, x_n^{(k)})$ to the estimate $(x_1^{(k+1)}, x_2^{(k+1)}, \cdots, x_n^{(k+1)})$ by searching for the maximum of the objective function along the direction having the components:

$$\delta_i^{(k)} = \sum_{j=1}^{n} H_{ij}^{(k)} g_j^{(k)}, \qquad i = 1, 2, \cdots, n, \tag{26}$$

where $H^{(k)}$ is a positive definite matrix which is chosen with the intention of providing a good rate of convergence. The method uses only values and first derivatives of the objective function, and it accounts for second derivative terms by changing $H^{(k)}$ according to differences between the calculated gradients of $F(x_1, x_2, \cdots, x_n)$. Usually the initial matrix $H^{(1)}$ is chosen to be the unit matrix, but any other positive definite matrix may be used instead, except that its elements should not conflict severely with the scaling of the variables $x_1, x_2, \cdots, x_n$, because of the effect of computer roundoff errors [3].

To calculate $H^{(k+1)}$, the Davidon iteration adds a correction term to the matrix $H^{(k)}$, that depends on the two vectors:

$$\begin{aligned}\sigma^{(k)} &= (x_1^{(k+1)}, x_2^{(k+1)}, \cdots, x_n^{(k+1)}) - (x_1^{(k)}, x_2^{(k)}, \cdots, x_n^{(k)}),\\ \gamma^{(k)} &= (g_1^{(k+1)}, g_2^{(k+1)}, \cdots, g_n^{(k+1)}) - (g_1^{(k)}, g_2^{(k)}, \cdots, g_n^{(k)}).\end{aligned} \tag{27}$$

This correction term is intended to make $H^{(k+1)}$ close to the matrix $-G^{-1}$, in order that the direction (26) shall be near to the Newton-Raphson direction (7). The use of the vectors (27) is guided by the fact that, if the objective function is quadratic, then the equation

$$\sigma^{(k)} = G^{-1}\gamma^{(k)} \tag{28}$$

holds, for $H^{(k+1)}$ is calculated to satisfy the equation

$$\sigma^{(k)} = -H^{(k+1)}\gamma^{(k)}. \tag{29}$$

Specifically the formula

$$H^{(k+1)} = H^{(k)} - \frac{\sigma^{(k)}\sigma^{(k)T}}{\sigma^{(k)T}\gamma^{(k)}} - \frac{H^{(k)}\gamma^{(k)}\gamma^{(k)T}H^{(k)}}{\gamma^{(k)T}H^{(k)}\gamma^{(k)}} \tag{30}$$

is used, where the superscript T indicates a row vector. Note that nearly all the other descriptions of Davidon's method treat the minimization, rather than the maximization, problem, in which case the signs of the first part of the correction term, and of the right-hand side of (29), are different.

This iteration has two important properties [14]. The first is that if $H^{(k)}$ is positive definite, then the matrix $H^{(k+1)}$ is also positive definite, and the second is that when the algorithm is applied to a quadratic function the directions of search are mutually conjugate. Therefore the method usually converges quickly. However, very occasionally the iterations make slow progress when the vectors $(x_1^{(k)}, x_2^{(k)}, \cdots, x_n^{(k)})$ are far from the solution, and sometimes this happens when a steepest ascent step would cause a substantial increase in the value of the objective function. Although in these cases the algorithm often manages to overcome the difficulty automatically, it is worrying that the inefficiency occurs. This remark typifies the fact that we do not really understand the algorithm, except in the trivial case when the objective function is quadratic.

Stewart [36] has extended Davidon's method, so that derivatives of the objective function are not needed. Instead the first derivatives are approximated by differences between function values, the main feature of Stewart's algorithm being that the intervals used in the difference approximations are chosen automatically, with a view to balancing truncation and roundoff errors. Numerical examples show that the resultant procedure is a little more efficient than Powell's [29] method when the number of variables is less than five, and for larger values of n the improvement is considerable.

8. Rank-one methods. This section describes and discusses just one recent idea that is likely to yield faster and more reliable algorithms for solving the unconstrained optimization problem, when values and first derivatives of the objective function are available. It has been stated by Broyden [7] and by Davidon [10], but

they missed a property that makes the technique especially promising. This property was noticed by Wolfe [38].

The idea is a development of Davidon's variable metric algorithm, for it is derived by considering definitions of $H^{(k+1)}$ different from (30), that also satisfy (29). In particular there is just one way of calculating $H^{(k+1)}$ so that the difference $\{H^{(k+1)} - H^{(k)}\}$ is a symmetric matrix of rank one. It is the formula:

$$H^{(k+1)} = H^{(k)} - \frac{\{\boldsymbol{\sigma}^{(k)} + H^{(k)}\boldsymbol{\gamma}^{(k)}\}\{\boldsymbol{\sigma}^{(k)} + H^{(k)}\boldsymbol{\gamma}^{(k)}\}^T}{\{\boldsymbol{\sigma}^{(k)} + H^{(k)}\boldsymbol{\gamma}^{(k)}\}^T\boldsymbol{\gamma}^{(k)}}, \tag{31}$$

and the idea is to calculate $H^{(k+1)}$ from $H^{(k)}$ by (31), where $\boldsymbol{\sigma}^{(k)}$ and $\boldsymbol{\gamma}^{(k)}$ are defined by (27).

To use (31) in an algorithm it is necessary to fix rules for calculating $(x_1^{(k+1)}, x_2^{(k+1)}, \cdots, x_n^{(k+1)})$ from $(x_1^{(k)}, x_2^{(k)}, \cdots, x_n^{(k)})$, and no doubt these rules will use the matrix $H^{(k)}$. The reason the idea is so valuable is that there are very many choices of rules such that n applications of (31) cause H to equal $-G^{-1}$ when the objective function is quadratic, and, because of the form of the Newton-Raphson iteration (7), this property can lead to fast convergence in the general case.

Both Broyden [7] and Davidon [10] follow (26) to define the rules for calculating $(x_1^{(k+1)}, x_2^{(k+1)}, \cdots, x_n^{(k+1)})$. They use the formula

$$x_i^{(k+1)} = x_i^{(k)} + \alpha^{(k)} \sum_{j=1}^{n} H_{ij}^{(k)} g_j^{(k)}, \tag{32}$$

where $\alpha^{(k)}$ is a parameter that is independent of i. Broyden proves that, if the choice of $\alpha^{(k)}$ is arbitrary, except that it must not cause $H^{(k+1)}$ to be singular or ill-defined (through the denominator of (31) being equal to zero), and if $F(x_1, x_2, \cdots, x_n)$ is quadratic, then $H^{(n+1)}$ will equal $-G^{-1}$. The important feature of this theorem is that it does not depend on calculating $\alpha^{(k)}$ by applying a one-dimensional search to maximize the objective function.

Davidon's new algorithm always uses $\alpha^{(k)} = 1$, so it can happen that expression (31) is not defined. However, if the vectors $\boldsymbol{\sigma}^{(k)}$ and $\boldsymbol{\gamma}^{(k)}$ are such that $H^{(k+1)}$ would not be positive definite, then Davidon defines $H^{(k+1)}$ by adding a different multiple of $\{\boldsymbol{\sigma}^{(k)} + H^{(k)}\boldsymbol{\gamma}^{(k)}\}\{\boldsymbol{\sigma}^{(k)} + H^{(k)}\boldsymbol{\gamma}^{(k)}\}^T$ to $H^{(k)}$, so that positive definiteness is obtained. Another feature of the method is that if $F(x_1^{(k+1)}, x_2^{(k+1)}, \cdots, x_n^{(k+1)})$ is less than $F(x_1^{(k)}, x_2^{(k)}, \cdots, x_n^{(k)})$, then, after $H^{(k+1)}$ has been calculated, $(x_1^{(k+1)}, x_2^{(k+1)}, \cdots, x_n^{(k+1)})$ is replaced by $(x_1^{(k)}, x_2^{(k)}, \cdots, x_n^{(k)})$ for the purposes of the next iteration.

The fact that linear searches are avoided is very valuable, because conjugate direction methods use many evaluations of the objective function just to locate optima along lines sufficiently accurately. Moreover, Box [5] shows by example that objective functions with well-defined solutions can be such that there are no optimal points along certain lines in the space of the variables, so the presence of linear searches can itself cause failure.

Wolfe's important observation [38] is that (31) can cause $H^{(n+1)}$ to equal $-G^{-1}$ in the quadratic case, without the restriction that $x_i^{(k+1)}$, $i = 1, 2, \cdots, n$, shall be calculated by (32). This statement is justified by the following argument.

If $F(x_1, x_2, \cdots, x_n)$ is the negative definite quadratic function (15), and if $\boldsymbol{\sigma}^{(k)}$ and $\boldsymbol{\gamma}^{(k)}$ are related by (27), then, for all vectors $\mathbf{u}$, the equation

$$\{\boldsymbol{\sigma}^{(k)} + H^{(k)}\boldsymbol{\gamma}^{(k)}\}^T\mathbf{u} = \boldsymbol{\gamma}^{(k)T}\{G^{-1} + H^{(k)}\}\mathbf{u} \tag{33}$$

holds. Therefore if $\boldsymbol{\sigma}^{(k)}$ and $\boldsymbol{\gamma}^{(k)}$ are such that (31) does not have a zero denominator, and if $\mathbf{u}$ satisfies the equation

$$G^{-1}\mathbf{u} = -H^{(k)}\mathbf{u}, \tag{34}$$

then we deduce, from (31) and (33), that $\mathbf{u}$ also satisfies the equation

$$G^{-1}\mathbf{u} = -H^{(k+1)}\mathbf{u}. \tag{35}$$

But for $j = 1, 2, \cdots, k$, the matrix $H^{(j+1)}$ is calculated to make the equation:

$$G^{-1}\boldsymbol{\gamma}^{(j)} = -H^{(j+1)}\boldsymbol{\gamma}^{(j)} \tag{36}$$

hold, so from our remarks we obtain the result:

$$G^{-1}\boldsymbol{\gamma}^{(j)} = -H^{(k+1)}\boldsymbol{\gamma}^{(j)}, \qquad j = 1, 2, \cdots, k. \tag{37}$$

It follows that if, for $k = 1, 2, \cdots, n$, the vectors $\boldsymbol{\sigma}^{(k)}$ are independent (which implies the independence of $\boldsymbol{\gamma}^{(k)}$, $k = 1, 2, \cdots, n$), and if these vectors are calculated so that (31) is well-defined, then $H^{(n+1)}$ is equal to $-G^{-1}$. Thus all the second derivative information is obtained without any linear searching, and we note that there is much freedom in the choice of $\boldsymbol{\sigma}^{(k)}$, $k = 1, 2, \cdots, n$.

Some ways of exploiting this idea are suggested in § 10.

Note that n applications of (31) is not the same as solving a large system of linear equations to determine the coefficients of a quadratic function, because solving the equations does not automatically cause the matrix $H^{(n+1)}$ to be symmetric.

9. Poor initial estimates. If the initial vector of variables $(x_1^{(1)}, x_2^{(1)}, \cdots, x_n^{(1)})$ is far from the position of the maximum of $F(x_1, x_2, \cdots, x_n)$, and if a conjugate direction algorithm is applied to solve the optimization algorithm, then it usually happens that most of the computer time is spent on obtaining a point that is close to the solution, rather than on completing the calculation after a good estimate has been obtained. Therefore the behavior of methods when $(x_1^{(k)}, x_2^{(k)}, \cdots, x_n^{(k)})$ is far from a maximum is of interest, but it has been studied very little. Some progress has been made by Greenstadt [21], and by Goldfeld, Quandt and Trotter [18], for they have suggested algorithms that, unlike the Newton-Raphson method, will act sensibly when the estimate $(x_1^{(k)}, x_2^{(k)}, \cdots, x_n^{(k)})$ is so poor that the matrix of second derivatives of the objective function is not negative definite. We consider the two methods in this section.

Greenstadt's method [21] is a simple extension of the Newton-Raphson method. The kth iteration calculates the second derivative matrix $G^{(k)}$ at the estimate $(x_1^{(k)}, x_2^{(k)}, \cdots, x_n^{(k)})$, and then an eigenvalue analysis obtains the reduction

$$G^{(k)} = \Omega\Lambda\Omega^T, \tag{38}$$

where Ω is orthogonal and Λ is diagonal. In place of G^{-1} in (7), Greenstadt uses the matrix $(\Omega\Lambda^*\Omega^T)^{-1}$ to calculate the search direction $(\delta_1^{(k)}, \delta_2^{(k)}, \cdots, \delta_n^{(k)})$, where Λ^* is the diagonal matrix obtained by replacing the diagonal elements of Λ by the

negatives of their moduli. The estimate $(x_1^{(k+1)}, x_2^{(k+1)}, \cdots, x_n^{(k+1)})$ is calculated by searching from $(x_1^{(k)}, x_2^{(k)}, \cdots, x_n^{(k)})$ in the direction $(\delta_1^{(k)}, \delta_2^{(k)}, \cdots, \delta_n^{(k)})$ for the maximum of the objective function. Although the original description is not supported by numerical examples, it is stated that the method works very well in practice. Conversations with various people have confirmed this assertion.

We have already met the main idea of the algorithm due to Goldfeld, Quandt and Trotter [18], for it is expressed by (12). The nonnegative parameter α is calculated so that the matrix $[\alpha I - G]$ is positive definite, the actual value of α being equal to the maximum of zero and R plus the largest eigenvalue of G, where R is a positive number that is adjusted automatically. One purpose of R is to control the length of the correction vector $(\delta_1, \delta_2, \cdots, \delta_n)$ (calculated by (12)), so that the value of the objective function at $(x_1^{(k)} + \delta_1^{(k)}, x_2^{(k)} + \delta_2^{(k)}, \cdots, x_n^{(k)} + \delta_n^{(k)})$ is larger than $F(x_1^{(k)}, x_2^{(k)}, \cdots, x_n^{(k)})$.

The need to calculate second derivatives and the need to solve the eigenvalue problem both detract from the utility of the above two algorithms. However, because of the lack of sensible strategies when the estimate $(x_1^{(k)}, x_2^{(k)}, \cdots, x_n^{(k)})$ is far from the solution, it can be worthwhile to study such methods. In particular, by experimenting with different algorithms that require second derivatives, efficient ideas may be identified, and only then may it be worthwhile to spend much ingenuity on avoiding the explicit calculation of G.

A good theoretical reason for their method is given by Goldfeld, Quandt and Trotter [18], and the same reason was noticed independently and earlier by Marquardt [26]. It applies when the objective function is quadratic, and when α exceeds the greatest eigenvalue of G. In this case the correction $(\delta_1, \delta_2, \cdots, \delta_n)$, defined by (12), has the property that $F(x_1 + \delta_1, x_2 + \delta_2, \cdots, x_n + \delta_n)$ is the greatest value of the objective function within the sphere of points whose Euclidean distance from $(x_1, x_2, \cdots, x_n)$ does not exceed $(\delta_1^2 + \delta_2^2 + \cdots + \delta_n^2)^{1/2}$. By choosing positive definite matrices that are different from I in (12), we can calculate best points within ellipsoids, rather than within spheres. These extensions are discussed in [19].

10. Conclusions. We have traced the main ideas of unconstrained optimization, with the intention of identifying successful algorithms, and of setting the scene for future research. Our main conclusions on the algorithms that have been in use for a number of years are that occasionally a direct search method is preferable, but usually it is better to apply Davidon's method [9] (if first derivatives are available), Powell's method [29] or Stewart's method [36] (if the user prefers not to program the calculation of derivatives). The number of variables that can be handled by these techniques depends strongly on $F(x_1, x_2, \cdots, x_n)$, a rough guide being that Davidon's algorithm has been successful for $n = 100$, and the other two methods have solved problems with twenty variables. The numerical evidence suggests that Stewart's technique becomes much more efficient than Powell's as the number of variables increases. However, if $F(x_1, x_2, \cdots, x_n)$ is not differentiable the suitability of conjugate direction methods is doubtful. In this case it is often necessary to study the properties of the objective function, in order to identify special features that will help locate the solution; but to avoid unnecessary effort it is prudent to try an automatic search method first, just in case it is successful.

Our conclusion on the theory of the algorithms is that a great deal of work needs to be done. For instance, in the cases when Davidon's method makes slow progress (although the gradient of $F(x_1, x_2, \cdots, x_n)$ is substantial), and then manages to take large steps again, it is not known whether the improvement is due to computer rounding errors, or whether it would occur if exact arithmetic were used. Also conditions for convergence and rates of convergence for nearly all the algorithms described in §§ 6–9 have not been found. It is hoped that this lack of theory will stimulate some research.

The most exciting prospect is the potential of the rank-one methods, described in § 8, because very many choices of displacements $(x_1^{(k+1)} - x_1^{(k)}, x_2^{(k+1)} - x_2^{(k)}, \cdots, x_n^{(k+1)} - x_n^{(k)})$ provide the exact solution when the objective function is quadratic. For instance, this freedom can be used to ensure that the successive displacements do not tend to lie in just a part of the full space of the variables, which is necessary to cause $H^{(k)}$ to converge to G^{-1}. Also this freedom permits Levenberg's idea [25], given by (12), to be exploited. Since these benefits can probably be obtained without the need for linear searches, it is likely that the current methods of unconstrained optimization will be replaced by much better algorithms.

REFERENCES

[1] HIROTUGU AKAIKE, *On a successive transformation of probability distribution and its application to the analysis of the optimum gradient method,* Ann. Inst. Statist. Math. Tokyo, 11 (1959), pp. 1–16.

[2] ROBERT M. BAER, *Note on an extremum locating algorithm,* Comput. J., 5 (1962), p. 193.

[3] YONATHON BARD, *On a numerical instability of Davidon-like methods*, IBM New York Scientific Center Rep. no. 320–2913, 1967.

[4] A. D. BOOTH, *Numerical Methods*, Butterworths, London, 1957.

[5] M. J. BOX, *A comparison of several current optimization methods, and the use of transformations in constrained problems*, Comput. J., 9 (1966), pp. 67–77.

[6] M. J. BOX, D. DAVIES AND W. H. SWANN, *Optimization Techniques*, I.C.I. Monograph no. 5, Oliver and Boyd Ltd., Edinburgh and London, 1968, to appear.

[7] C. G. BROYDEN, *Quasi-Newton methods and their application to function minimization,* Math. Comp., 21 (1967), pp. 368–381.

[8] A. CAUCHY, *Methode générale pour la résolution des systemes d'équations simultanées*, C.R. Acad. Sci. Paris, 25 (1847), p. 536.

[9] W. C. DAVIDON, *Variable metric method for minimization*, A.E.C. Research and Development Rep. ANL-5990 (rev.), 1959.

[10] ———, *Variance algorithm for minimization*, Comput. J., 10 (1968), pp. 406–410.

[11] A. W. DICKINSON, *Nonlinear optimization: some procedures and examples*, Proc. 19th ACM National Conference, 1964, pp. E1.2/1–E1.2/8.

[12] D. K. FADEEV AND V. N. FADEEVA, *Computational Methods of Linear Algebra*, W. H. Freeman, San Francisco, 1963.

[13] R. FLETCHER, *Function minimization without evaluating derivatives—a review*, Comput. J., 8 (1965), pp. 33–41.

[14] R. FLETCHER AND M. J. D. POWELL, *A rapidly convergent descent method for minimization*, Ibid., 6 (1963), pp. 163–168.

[15] R. FLETCHER AND C. M. REEVES, *Function minimization by conjugate gradients,* Ibid., 7 (1964), pp. 149–154.

[16] G. E. FORSYTHE, *On the asymptotic directions of the s-dimensional optimum gradient method,* Numer. Math., 11 (1968), pp. 57–76.

[17] G. E. FORSYTHE AND T. S. MOTZKIN, *Asymptotic properties of the optimum gradient method,* Bull. Amer. Math. Soc., 57 (1951), p. 183.

[18] S. M. GOLDFELD, R. E. QUANDT AND H. F. TROTTER, *Maximization by quadratic hill-climbing*, Econometrica, 34 (1966), pp. 541–551.

[19] ———, *Maximization by improved quadratic hill-climbing and other methods*, Econometric Research Memo. 95, Princeton University, 1968.
[20] A. A. GOLDSTEIN, *Cauchy's method of minimization*, Numer. Math., 4 (1962), pp. 146–150.
[21] JOHN GREENSTADT, *On the relative efficiencies of gradient methods*, Math. Comp., 21 (1967), pp. 360–367.
[22] M. R. HESTENES AND E. STIEFEL, *Methods of conjugate gradients for solving linear systems*, J. Res., Nat. Bur. Standards, 49 (1952), pp. 409–436.
[23] R. HOOKE AND T. A. JEEVES, *Direct search solution of numerical and statistical problems*, J. Assoc. Comput. Mach., 8 (1961), pp. 212–221.
[24] ALBERTO LEON, *A comparison among eight known optimizing procedures*, Recent Advances in Optimization Techniques, A. Lavi and T. P. Vogl, eds., John Wiley, New York, 1966.
[25] K. LEVENBERG, *A method for the solution of certain non-linear problems in least squares*, Quart. Appl. Math., 2 (1944), pp. 164–168.
[26] DONALD W. MARQUARDT, *An algorithm for least-squares estimation of nonlinear parameters*, J. Soc. Indust. Appl. Math., 11 (1963), pp. 431–441.
[27] J. A. NELDER AND R. MEAD, *A simplex method for function minimization*, Comput. J., 7 (1965), pp. 308–313.
[28] M. J. D. POWELL, *An iterative method for finding stationary values of functions of several variables*, Ibid., 5 (1962), pp. 147–151.
[29] ———, *An efficient method for finding the minimum of a function of several variables without calculating derivatives*, Ibid., 7 (1964), pp. 155–162.
[30] ———, *Minimization of functions of several variables*, Numerical Analysis: An Introduction, J. Walsh, ed., Academic Press, London, 1966, pp. 143–157.
[31] H. H. ROSENBROCK, *An automatic method for finding the greatest or the least value of a function*, Comput. J., 3 (1960), pp. 175–184.
[32] B. V. SHAH, R. J. BEUHLER AND O. KEMPTHORNE, *Some algorithms for minimizing a function of several variables*, J. Soc. Indust. Appl. Math., 12 (1964), pp. 74–92.
[33] C. S. SMITH, *The automatic computation of maximum likelihood estimates*, Rep. S.C. 846/MR/40, N.C.B. Scientific Dept., 1962.
[34] H. A. SPANG, III, *A review of minimization techniques for nonlinear functions*, this Review, 4(1962), pp. 343–365.
[35] W. SPENDLEY, G. R. HEXT AND F. R. HIMSWORTH, *Sequential application of simplex designs in optimization and evolutionary operation*, Technometrics, 4 (1962), p. 441.
[36] G. W. STEWART, III, *A modification of Davidon's minimization method to accept difference approximations of derivatives*, J. Assoc. Comput. Mach., 14 (1967), pp. 72–83.
[37] W. H. SWANN, *Report on the development of a new direct search method of optimization*, Research Note 64/3, Central Instrument Laboratory, I.C.I. Ltd., 1964.
[38] P. WOLFE, *Another variable metric method,* Working paper, 1967.
[39] WILLARD I. ZANGWILL, *Minimizing a function without calculating derivatives*, Comput. J., 10 (1967), pp. 293–296.
[40] G. ZOUTENDIJK, *Methods of Feasible Directions*, Elsevier, Amsterdam, 1960.