JOURNAL OF OPTIMIZATION THEORY AND APPLICATIONS: Vol. 18, No. 2. FEBRUARY 1976

New Least-Square Algorithms[1]

W. C. DAVIDON[2]

Communicated by D. R. Fulkerson

Abstract. New algorithms are presented for approximating the minimum of the sum of squares of M real and differentiable functions over an N-dimensional space. These algorithms update estimates for the location of a minimum after each one of the functions and its first derivatives are evaluated, in contrast with other least-square algorithms which evaluate all M functions and their derivatives at one point before using any of this information to make an update. These new algorithms give estimates which fluctuate about a minimum rather than converging to it. For many least-square problems, they give an adequate approximation for the solution more quickly than do other algorithms.

Key Words. Least-square methods, variable-metric methods, gradient methods, nonlinear programming.

1. Introduction

New algorithms have been developed and tested for minimizing the sum of squares of M real and differentiable functions ϕ_m with a common N-dimensional domain. These algorithms update estimates for the location of a minimum after each function ϕ_m and its first derivatives are evaluated, in contrast with other least-square algorithms in which all M functions and their derivatives are evaluated at one point before any of this information is used to make an update. This more frequent updating often saves considerable computation time, otherwise spent gathering and soon discarding information about the behavior of the functions in regions far from the eventual minimum. This approach is advantageous in those situations (typical case: adjusting N theoretical parameters to fit M experimental data points) in which the following conditions are met:

[1] It is a pleasure to thank J. Chesick of Haverford College for suggesting and implementing the application of this algorithm to x-ray crystallography.

[2] Professor of Physics, Haverford College, Haverford, Pennsylvania.

(i) $M \gg N$, i.e., there are many more data points than parameters;

(ii) rapid approach to an approximate minimum is more important than final convergence to an exact minimum;

(iii) for many choices of weighting factors λ_m, the location of the minimum for the weighted sum of squares $\sum_m \phi_m^2 \lambda_m$ closely approximates that for the sum $\sum_m \phi_m^2$.

The presentation of these algorithms and their comparison with others is facilitated by using a common mode of description for them all. At the start of their ith iteration, information previously gathered about the functions ϕ_m and their derivatives is used to construct a quadratic approximation f_i to the sum $\sum_m \phi_m^2$ to be minimized. In the ith iteration, the values of one or more of the functions ϕ_m and their derivatives are determined at the point x_i where the quadratic approximation f_i has its minimum. This additional information is used to update the quadratic approximation from f_i to f_{i+1} to start the next iteration.

As long as each quadratic approximation f_i has a unique minimum, it can be expressed as

$$f_i(x) = \alpha_i + (x - x_i)^T H_i^{-1} (x - x_i), \tag{1}$$

where α_i is the minimum value for f_i, x_i is the $N \times 1$ column vector giving the location of this minimum, and H_i is the $N \times N$ positive-definite matrix whose inverse is the Hessian matrix of second derivatives of $f_i/2$. The superscript T denotes matrix transposition.

In the Newton–Raphson algorithm (Ref. 1), the quadratic approximation f_{i+1} is obtained by truncating the Taylor expansion for $\sum_m \phi_m^2$ about the point x_i, where the preceding quadratic approximation f_i has its minimum. In the Gauss algorithm (Ref. 2), f_{i+1} is obtained instead by summing the squares of first-degree approximations to the ϕ_m:

$$f_{i+1}(x) = \sum_m (\phi_m(x_i) + (x - x_i)^T \nabla \phi_m(x_i))^2, \tag{2}$$

where $\nabla \phi_m$ is the $N \times 1$ gradient vector whose components are the first derivatives of ϕ_m. The approximation f_{i+1} of the Levenberg (Ref. 3) and Marquardt (Ref. 4) algorithms differs from that of Eq. (2) by a positive multiple of $|x - x_i|^2$.

In all of these algorithms, the locations x_i of the minima of the quadratic approximations satisfy

$$x_{i+1} = x_i - H_{i+1} [\sum_m \phi_m \nabla \phi_m] x_i, \tag{3}$$

where the $N \times N$ matrices H_i are defined by Eq. (1) and are obtained by matrix inversion.

In the new algorithms presented here, the quadratic approximation f_{i+1} is obtained by adding just one term from the sum on the right of Eq. (2) to a

multiple of the previous quadratic approximation, i.e.,

$$f_{i+1}(x) = [\phi_{m(i)}(x_i) + (x - x_i)^T \nabla \phi_{m(i)}(x_i)]^2 + \lambda f_i(x), \tag{4}$$

where λ is a positive number less than one. In this case, H_{i+1} can be obtained by updating H_i, without the need for matrix inversion. The location x_{i+1} of the minimum of f_{i+1} is given by

$$x_{i+1} = x_i - H_{i+1}[\phi_{m(i)} \nabla \phi_{m(i)}] x_i, \tag{5}$$

instead of by Eq. (3).

If the functions ϕ_m are all linear, if $f_0 = 0$, and if $\lambda = 1$, then the Mth quadratic approximation f_M, defined by Eq. (4), is just $\sum_m \phi_m^2$, so that x_M is then the exact solution of the linear least-square problem. In general, however, the sequence of x_i does not converge to the location of the minimum of $\sum_m \phi_m^2$ but fluctuates about it with an amplitude that depends on λ, as determined in Section 3.

A physical analog for the behavior of these algorithms when searching through nonlinear regions is given by different measurement processes on a fluctuating, time-dependent quantity, e.g., radiation from a changing radioactive source. In the analog of the traditional least-square algorithms, a summation is done over blocks of time of fixed duration. The output available at any time, analogous to the location x_i at each iteration, is based only on the last completed block. In the analog of these new algorithms, the output at any time is instead a weighted sum over past times with a weighting factor which decreases exponentially with time delay. As a result, new information affects the output promptly and is not delayed until the completion of the next full block.

Throughout this paper, we consider only real numbers and matrices. Except for certain traditional symbols, we use lower case Greek letters for numbers other than integers, lower case italic letters for column vectors, and upper case italic letters for square matrices.

2. Algorithms

The basic algorithm for updating the number α_i, the $N \times 1$ column vector x_i, and the $N \times N$ positive definite matrix H_i which together specify each quadratic approximation f_i by Eq. (1) is

$$\alpha_{i+1} = (\alpha_i + \phi_i^2/\gamma_i)\lambda, \tag{6}$$

$$x_{i+1} = x_i - H_i \phi_i \nabla \phi_i / \gamma_i, \tag{7}$$

$$H_{i+1} = [H_i - H_i \nabla \phi_i (H_i \nabla \phi_i)^T / \gamma_i]/\lambda, \tag{8}$$

where

$$\phi_i = \phi_{m(i)}(x_i) \qquad \text{and } \gamma_i = \lambda + (\nabla\phi_i)^T H_i \nabla\phi_i.$$

In the next section, we prove that Eqs. (6)–(8) are necessary and sufficient for the quadratic approximations given by Eq. (1) to satisfy Eq. (4). Equation (6) for updating the minimum values α_i of the f_i is not needed for the search itself but is useful in some criteria for termination. For example, it is often satisfactory to stop when a specified number of successive iterations fail to decrease the smallest value yet obtained for α.

The sequence in which the M functions ϕ_m and their gradients $\nabla\phi_m$ are evaluated is determined by $m(i)$ and is quite arbitrary, provided that all of them are evaluated once in every M consecutive iterations. Fluctuations can usually be reduced by *scrambling* this sequence, so that consecutive iterations are more likely to be representative, for example, by choosing

$$m(i) = (ip)_{\text{mod}M},$$

where p is a prime.

The weighting factor λ need not be kept the same for all iterations. In one variation, $\lambda_i = 1$ for all i not divisible by M and every Mth λ is a positive number much less than one. In this case, after each cycle in which all the ϕ_m and their gradients $\nabla\phi_m$ are evaluated, the resulting α_i is a good approximation to the sum of squares $\sum_m \phi_m^2$. Another variation of this algorithm is to start with one value for λ, typically $N/(N+1)$, and then increase it towards one as the search progresses. In this way, a rough approximation for the minimum is reached more quickly and then increasing λ decreases the fluctuations about it.

Instead of storing and updating H directly, there are computational advantages to representing it in terms of other matrices. One alternative,

$$H = A/\sigma,$$

reduces the number of arithmetic operations needed in each iteration, since multiplying σ by λ is equivalent to the N^2 multiplications needed to scale H directly by the factor $1/\lambda$. In this way, the number of multiplications or divisions needed in each iteration to update x_i and H_i is $(3N^2+7N+2)/2$. We note that, for large N, the Gauss least-square algorithm requires approximately $N^2/2$ multiplications after each evaluation of a function ϕ_m and its gradient, and then approximately $N^3/2$ after each cycle in which all these functions are evaluated in order to invert the Hessian of the quadratic approximation. Hence, if M is comparable to N, this algorithm requires more multiplications than Gauss's by a factor of $3/2$.

A representation for the matrix H which reduces the effects of rounding is

$$H = JJ^T/\sigma,$$

where J is a matrix which need not be symmetric. This representation requires approximately twice the number of multiplications per iteration, $3(N^2+N+1)$, and also doubles the required storage space, since H and A (but not J) are symmetric. However, it ensures that $H=JJ^T/\sigma$ stays positive regardless of rounding. When this representation is used, the matrix J and scaling factor σ are updated by $\sigma_{i+1}=\sigma_i\lambda$, $J_{i+1}=J_i-J_ik_ik_i^T/(\rho_i+(\rho_i\sigma_{i+1})^{1/2})$, where

$$k_i=J_i^T\nabla\phi_i \qquad \text{and } \rho_i=\sigma_{i+1}+k_i^Tk_i.$$

With

$$H_i=J_iJ_i^T/\sigma_i,$$

this update is equivalent to Eq. (8).

3. Properties of the Updates

Theorem 3.1. For given i, let α_i, ϕ_i, and λ be any three numbers with $\lambda>0$; let x_i and $\nabla\phi_i$ be any $N\times 1$ column vectors; and let H_i be any $N\times N$ positive-definite matrix. Define the quadratic function f_i of x by Eq. (1) and f_{i+1} by

$$f_{i+1}(x)=(\phi_i+(x-x_i)^T\nabla\phi_i)^2+\lambda f_i(x). \tag{9}$$

Then, f_{i+1} is of the form of Eq. (1) for some α_{i+1}, x_{i+1}, and H_{i+1}, and these are given uniquely by Eqs. (6), (7), and (8), respectively.

Proof. Substitute $f_i(x)$ from Eq. (1) into Eq. (9) to obtain

$$f_{i+1}(x)=\lambda\alpha_i+\phi_i^2+2\phi_i(x-x_i)^T\nabla\phi_i+(x-x_i)^T(\lambda H_i^{-1}+\nabla\phi_i(\nabla\phi_i)^T)(x-x_i).$$

Since two polynomials in $x-x_i$ are equal for all x iff corresponding coefficients are equal, f_{i+1} is given by Eq. (1) for some α_{i+1}, x_{i+1}, and H_{i+1} iff

$$\alpha_{i+1}+(x_i-x_{i+1})^TH_{i+1}^{-1}(x_i-x_{i+1})=\lambda\alpha+\phi_i^2, \tag{10}$$

$$H_{i+1}^{-1}(x_i-x_{i+1})=\phi_i\nabla\phi_i, \tag{11}$$

$$H_{i+1}^{-1}=\lambda H_i^{-1}+\nabla\phi_i(\nabla\phi_i)^T. \tag{12}$$

Since

$$(X^{-1}+yz^T)^{-1}=X-Xyz^TX/(1+z^TXy)$$

for any invertible matrix X and column vectors y and z with

$$1+z^TXy\neq 0,$$

Eq. (12) is satisfied iff H_{i+1} is given by Eq. (8). Note that H_{i+1} is positive definite, since $\lambda > 0$ and H_i is positive definite. Equation (7) then follows from Eq. (11) and Eq. (6) from Eq. (10), completing the proof.

From this theorem, it follows immediately by induction that, when the functions ϕ_m are all linear,

$$f_M(x) = \phi^2_{m(M-1)}(x) + \lambda\phi^2_{m(M-2)}(x) + \cdots + \lambda^{M-1}\phi^2_{m(0)}(x) + \lambda^M f_0(x) \quad (13)$$

for all x. Since x_M is the location for the minimum of f_M, it follows that, if f_0 is zero and $\lambda = 1$, then $f_M = \sum_m \phi_m{}^2$ and x_M is the exact solution of the least-square problem. In general, the sequence of x_i does not converge to a minimum but fluctuates about it with a mean value $\langle x\rangle$ and variance matrix $\langle \Delta x \Delta x^T\rangle$, which we now estimate.

To simplify our analysis, we assume that the fluctuations in the successive values for the matrices H_i are small, so that

$$\langle H^{-1}\rangle \approx \langle H\rangle^{-1};$$

that the fluctuations in H_i are uncorrelated with those of $\phi_i \nabla\phi_i$, so that

$$\langle H\phi\nabla\phi\rangle \approx \langle H\rangle\langle\phi\nabla\phi\rangle;$$

and that the ϕ_i are approximately linear over the region of fluctuations. By averaging both sides of Eq. (5), we obtain

$$\langle H\phi\nabla\phi\rangle = 0,$$

which with our assumptions that

$$\langle H\phi\nabla\phi\rangle \approx \langle H\rangle\langle\phi\nabla\phi\rangle$$

and that $\langle H\rangle$ is invertible gives

$$\langle\phi\nabla\phi\rangle = 0.$$

Hence,

$$\left\langle \nabla \sum_m \phi_m{}^2/2 \right\rangle = \left\langle \sum_m \phi_m \nabla\phi_m \right\rangle = M\langle\phi\nabla\phi\rangle = 0.$$

Since the average gradient of the sum of squares vanishes, and since we assume that ϕ_m are linear over the region of fluctuations, it follows that the mean value $\langle x\rangle$ of the x_i is the approximate solution of the least-square problem.

To estimate the variance matrix $\langle \Delta x \Delta x^T\rangle$, we assume that successive steps $x_{i+1} - x_i$ are uncorrelated and that the average number of these which contribute to Δx is

$$1 + \lambda + \lambda^2 + \cdots = 1/(1-\lambda).$$

Together with Eq. (7), this gives

$$\langle \Delta x \Delta x^T \rangle = \langle (H\phi \nabla \phi)(H\phi \nabla \phi)^T \rangle / (1-\lambda).$$

Adding the assumption that the fluctuations in ϕ_i^2, H_i, and $\nabla \phi_i \nabla^T \phi_i$ are uncorrelated gives

$$\langle \Delta x \Delta x^T \rangle = \langle \phi^2 \rangle \langle H \rangle \langle \nabla \phi \nabla^T \phi \rangle \langle H \rangle / (1-\lambda).$$

It follows from Eqs. (1) and (13) that

$$\langle H^{-1} \rangle = \langle \nabla \phi \nabla^T \phi \rangle (1 + \lambda + \lambda^2 + \cdots) = \langle \nabla \phi \nabla^T \phi \rangle / (1-\lambda),$$

$$\langle \alpha \rangle = \langle \phi^2 \rangle (1 + \lambda + \lambda^2 + \cdots) = \phi^2 / (1-\lambda).$$

Combining these averages with the assumption that

$$\langle H^{-1} \rangle \approx \langle H \rangle^{-1}$$

gives

$$\langle \Delta x \Delta x^T \rangle = \langle \phi^2 \rangle \langle \nabla \phi \nabla^T \phi \rangle^{-1} (1-\lambda);$$

or, equivalently,

$$\langle \Delta x \Delta x^T \rangle = \langle \alpha \rangle \langle H \rangle (1-\lambda).$$

The amplitude of the fluctuations in the successive estimates x_i for the location of a minimum is proportional to the square root of the variance and, hence, to $(1-\lambda)^{1/2}$.

4. Numerical Results

In comparisons of least-square algorithms, a frequently used and machine-independent measure of calculation time has been an *equivalent function evaluation.* One of these corresponds to a calculation of the sum of squares of all M functions ϕ_m, and it is usual to consider the calculation of the N first derivatives of this sum as N equivalent function evaluations. To avoid possible confusion between an equivalent function evaluation and an evaluation of each one of the functions ϕ_m, we define a *data cycle* as an evaluation of all M functions ϕ_m and their first derivatives, without regard to the points at which they are evaluated. Thus, each iteration of other least-square algorithms constitutes one data cycle, while M iterations of these algorithms is one data cycle. When determining theoretical parameters which fit experimental data, all the data is considered just once in each data cycle (hence, the term). Usually, one data cycle corresponds to $N+1$ equivalent function evaluations.

These algorithms have been tested with several types of least-square problems: a linear but ill-conditioned one to check programming and sensitivity to rounding, several standard test problems for comparison with other algorithms, and an application in x-ray crystallography with two hundred parameters and several thousand data points.

Example 4.1. *Polynomial Curve Fitting*:

$$\phi_m(x) = \sum_{n=0}^{N-1} x_n m^n \qquad \text{for } 1 \leq m < M,$$

$$\phi_0(x) = x_0 - 1.$$

For $M \leq N$, exact solutions to the M simultaneous equations $\phi_m(x) = 0$ can be readily determined, e.g., by the method of Parker (Ref. 5), even though for large M the problem is ill-conditioned. Rounding errors can, therefore, be evaluated by comparing these exact solutions with the output of the algorithm in the limit as the initial H^{-1} goes to zero. This is the only problem considered in which representing H as JJ^T/σ significantly reduced rounding.

One set of input quantities used in testing the algorithm was: $M = N = 15$, $x = 0$ (the 15×1 null vector), $H = 10^{12}$ times the 15×15 unit matrix, $\lambda = 0{\cdot}7$, and $p = 7$, i.e., the data point selected in the ith iteration was given by

$$m(i) = (7i)_{\text{mod } 15}.$$

In less than two data cycles, the components of the parameter vector x were within 1% of the exact values, which rounded to six digits are:
1, -3.25156, 4.49833, -3.56728, 1.83213, -0.649971, 0.165286, -3.07583×10^{-2}, 4.22596×10^{-3}, -4.28000×10^{-4}, 3.15416×10^{-5}, -1.64404×10^{-6}, 5.74111×10^{-8}, -1.20443×10^{-9}, and 1.14707×10^{-11}.

Example 4.2. *Exponential Functions with Zero Residual* (Ref. 6):

$$\phi_m(x) = \exp(-x_0\eta_m) - \exp(-x_1\eta_m) - x_2[\exp(-\eta_m) - \exp(-10\eta_m)],$$

$$\eta_m = (m+1)/10 \qquad \text{and } 0 \leq m < 10.$$

This problem has a zero residual at $x = (1, 10, 1)$ and at all x with $x_0 = x_1$ and $x_2 = 0$. Among the sets of input quantities used was $x = (0, 10, 20)$, H equal to the 3×3 unit matrix, $\lambda = 0.7$, and $p = 7$. After 7 data cycles, the location of a minimum was approximated by

$$x = (0.99983, 10.001, 1.0001).$$

Almost the same approximation was also obtained in less than 8 data cycles with a starting location $x = (0, 20, 20)$.

This calculation time corresponds to about 30 equivalent function evaluations, compared to 20 for the Gauss algorithm under similar conditions, 100 for the Levenberg–Marquardt algorithm, and 140 for the Davidon–Fletcher–Powell algorithm (Refs. 7–8).

Example 4.3. *Nonlinear Problem with a Large Residual* (Ref. 9):

$$\phi_m(x) = [x_0 + x_1\eta_m - \exp(\eta_m)]^2 + (x_2 + x_3 \sin \eta_m - \cos \eta_m)^2,$$

$$\eta_m = (m+1)/5 \qquad \text{and } 0 \leq m < 20.$$

The minimum value for $\sum_m \phi_m^2(x)$ is approximately 85,800 at

$$x = (-11.59, 13.20, -0.4034, 0.2368).$$

Input quantities used were $x = (25, 5, -5, -1)$, H equal to the 4×4 unit matrix, $\lambda = 0.8$ and $p = 7$. After 4 data cycles, the location of the minimum was approximated by

$$x = (-11.59, 12.86, 1.747, -1.526).$$

While the algorithm itself does not compute the sum of squares of the ϕ_m at any one point, the value for this sum at the estimated minimum is 100, 124. Increasing the weighting factor λ from 0.8 to 0.9 decreases the fluctuations in successive approximations and improves the estimate for a location of a minimum to one at which the sum of squares is 87,339, but this increases the number of data cycles from 4 to 7, corresponding to about 20 and 35 equivalent function evaluations, respectively.

This particular example of a least-square problem with a large residual was chosen because it was used by Brown and Dennis to test their algorithm which uses changes in the gradients $\nabla\phi_m$ to estimate the M different $N \times N$ Hessian matrices of the ϕ_m which are all stored and updated. Their algorithm obtained a better estimate for the minimum but with somewhat more calculation, 50 equivalent function evaluations.

Example 4.4. *Application to X-Ray Crystallography* (Ref. 10). In this application, each of the M functions ϕ_m is the difference between an experimental data point and a theoretical prediction based on N theoretical parameters, divided by the standard deviation in the experimental measurement. Typical values for M and N are

$$2000 \leq M \leq 4000 \qquad \text{and } 100 \leq N \leq 250.$$

The functions ϕ_m are composed from exponential and trigonometric functions as well as rational functions of their N arguments.

The new algorithms have been compared with the Gauss algorithm. In both cases, to save time and memory, the parameters expected to be most correlated are grouped together, and zero is used for all elements of the $N \times N$ matrix H corresponding to pairs of parameters in different groups. The nonzero diagonal blocks in the resulting matrix are typically less than 20×20, so that, instead of tens of thousands of elements, only thousands are stored and updated.

In some cases, the initial estimates for the location of a minimum were good enough, so that both the Gauss algorithm and this one reached an adequate approximation in one or two data cycles. In other cases, when the Gauss algorithm used from three to fifteen data cycles to reach an acceptable approximation, this new algorithm typically required only about one-third as many cycles. There are also cases in which the Gauss algorithm diverged while this one did not, but none when this one diverged.

5. Conclusions

Analysis and tests of these new least-square algorithms show them to have the following features.

(i) Successive approximations for a minimum do not decrease monotonically but fluctuate about a decreasing average. Because of these fluctuations, there is no final convergence to an exact minimum; but, in many applications, an adequate approximation can be obtained more quickly than with other algorithms. While convergence properties of algorithms are of considerable mathematical interest, in many applications locating a small neighborhood of a minimum is of more significance and poses more difficulties than final convergence within this neighborhood.

(ii) No matrix is inverted in the algorithm, in contrast to the Newton–Raphson, Gauss, Levenberg–Marquardt, and other ones. Instead, each matrix H_{i+1} is obtained from its predecessor H_i by adding a certain rank-one matrix. The need for matrix inversion is eliminated, because here the Hessian matrix of second derivatives of each quadratic approximation f_{i+1} differs from that of f_i by a matrix of rank one. Not only does this avoid the rounding problems which arise when solving large and often ill-conditioned linear systems, but it makes possible the use of a singular matrix H to impose linear constraints. It follows directly from the updates specified in the algorithm that the location x_i of the minimum of the ith quadratic approximation f_i differs from x_0 by a vector in the column space of H_0. Hence, to impose constraints, the columns (and since it is symmetric, the rows as well) of H_0 are chosen to span just the subspace of displacements consistent with the constraints.

(iii) The choice of the weighting factor λ, or of strategies for changing it in the course of the search for a minimum, allows for matching the needs of a particular situation. Smaller values for λ in the range of 0 through 1 allow the search to adapt quickly to changes in the behavior of the function but introduce larger fluctuations. This may be advantageous in early stages of a search, while a λ closer to one causes more data points to contribute substantially to the weighted sum of squares being minimized, giving a slower response with smaller fluctuations.

References

1. RAPHSON, J., *Analysis Aequationum Universalis*, London, England, 1690.
2. GAUSS, K. F., *Theoria Motus Corporum Coelestium*, Werke 7, pp. 240–254, Königlichen Gesellschaft der Wissenschaften, Goettingen, Germany, 1809.
3. LEVENBERG, K., *A Method for the Solution of Certain Non-linear Problems in Least Squares*, Quarterly of Applied Mathematics, Vol. 2, pp. 164–168, 1944.
4. MARQUARDT, D. W., *An Algorithm for Least Squares Estimation*, SIAM Journal on Applied Mathematics, Vol. 11, pp. 431–441, 1963.
5. PARKER, F. D., *Inverses of Vandermonde Matrices*, American Mathematical Monthly, Vol. 71, pp. 410–411, 1964.
6. BOX, M. J., *A Comparison of Several Current Optimization Methods*, Computer Journal, Vol. 9, pp. 67–77, 1966.
7. BROWN, K., and DENNIS, J., *Derivative-Free Analogues of the Levenberg–Marquardt and Gauss Algorithms for Non-linear Least Squares Approximation*, IBM Philadelphia Scientific Center, Technical Report No. 320-2994, 1970.
8. FLETCHER, R., and POWELL, M. J. D., *A Rapidly Convergent Descent Method for Minimization*, Computer Journal, Vol. 6, pp. 163–168, 1963.
9. BROWN, K., and DENNIS, J., *New Computational Algorithms for Minimizing a Sum of Squares of Non-linear Functions*, Yale University, Department of Computer Science, Research Report No. 71-6, 1971.
10. CHESWICK, J., DAVIDON, W., and SCHOEN, C., *Application of a New Least Squares Method to Structure Refinement*, Acta Crystallographica (to appear).