OPTIMIZATION OF MANAGEMENT SYSTEMS BY THE CONJUGATE GRADIENT METHODS

by 6 P O

BALAKRISHNAN RADHAKRISHNAN NAIR

B.E., UNIVERSITY OF MADRAS

MADRAS, INDIA, 1959

A MASTER'S REPORT

submitted in partial fulfillment of the

requirements for the degree

MASTER OF SCIENCE

Department of Industrial Engineering

KANSAS STATE UNIVERSITY

Manhattan, Kansas

1969

Approved by

G. S. Hines Lee

Major Professor

TABLE OF CONTENTS

1. INTRODUCTION

The optimization of management systems frequently involves the maximization or minimization of non-linear systems of equations. While efficient algorithms have been developed for handling linear problems, no general algorithms exist for solving non-linear optimization problems. Various methods like the calculus of variations, the maximum principle and dynamic programming have been developed in recent years. However, only dynamic programming has been applied extensively to management problems [1]. The main limitation with this technique is that, owing to the dimensionality difficulties, it cannot be applied to problems with a large number of state variables. While several techniques have been devised to circumvent this limitation, these essentially involve trading computer memory for computing time and are not as efficient as the original dynamic programming algorithm.

The difficulties in obtaining analytical solutions to non-linear optimization problems have led to the development of a large number of direct methods of optimization. Many of them, like Rosenbrock's method [2], the pattern search method of Hooke and Jeeves [3,4], and the simplex method of Nelder and Mead [5], rely on an ad hoc rather than a theoretical approach to the problem [15]. Gradient methods, on the other hand, are classical in origin and based on the fact that if we move in the direction of the gradient of $f(x)$, we will move in the direction of the maximum rate of increase in $f(x)$ [6,7,13]. An iterative procedure is developed where a step is taken along the gradient and the new values of the function and gradient computed. The process is repeated until the

gradient assumes a sufficiently small value, showing that a stationary point has been reached. It has been well established [7] that such a method converges too slowly for practical applications. Hestenes and Stiefel [8] devised an ingenious scheme to accelerate the convergence of the gradient method. They used the accumulated knowledge of the behaviour of the function to generate new directions of search at the end of each iteration. This technique, called the conjugate-gradient method, is used by Fletcher and Reeves [9] to find the unconstrained minimum or maximum of a function of several variables.

All gradient methods require the gradient vector, or the vector of first partial derivatives of the function, to be evaluated. However, it is frequently the case that it is laborious or practically impossible to calculate these derivatives and there is a definite need for optimization techniques which do not require them. Powell [10] devised a method of finding the unconstrained minimum or maximum of a function, by searching along linearly independent directions, called conjugate directions, generated during each iteration cycle. The method does not require the evaluation of derivatives.

Both the methods of Fletcher and Reeves and Powell have the advantage of being able to handle problems with a fairly large number of state variables. They have the property of quadratic convergence, that is, if a quadratic function of n variables is being optimized, the optimum is reached in not more than n iterations. Both the methods can be easily programmed. The disadvantages are: (1) it cannot recognize constraints on the variables and (2) it may converge to a local extremum. The latter disadvantage can be partly overcome by using different starting values.

A comparison is made of the results obtained with each technique. Some conclusions are also drawn regarding the relative merits and demerits of the two methods.

2. THE CONJUGATE GRADIENT METHOD OF FLETCHER AND REEVES

Fletcher and Reeves [9] devised a method of locating the unconstrained minimum of a function of n variables whose value f(x) and the gradient vector g(x) can be calculated at any point x.

Assume that in the neighborhood of the required minimum h, the function may be expanded in the form

$$f(x) = f(h) + \frac{1}{2} (x - h)^T A (x - h) + \text{higher terms} \qquad (1)$$

where A, the matrix of second-order partial derivatives, is symmetric and positive-definite.

Virtually all iterative minimization procedures locate h as the limit of a sequence $x_0, x_1, x_2, \ldots$ where x_0 is an initial approximation to the position of the minimum. Also, for each $i \geq 0$, x_{i+1} is the position of the minimum of f(x) with respect to variations along the line through x_i in some specified direction ξ_i. Thus, for example, the method of steepest descent uses the direction of the negative gradient of f(x) at x_i, and the method of changing one variable at a time uses cyclically the directions of the n coordinate axes.

In the method of Fletcher and Reeves, the successive directions of search are A-conjugate, satisfying the condition

$$\xi_i^T A \xi_j = 0 \quad \text{for } i \neq j \qquad (2)$$

Setting $g(x_i) = g_i$ for each i, the step from x_i to x_{i+1} is determined by the relation

$$g_{i+1}^T \xi_i = 0 \qquad (3)$$

$$x_{i+1} = x_i + \lambda_i \xi_i \tag{4}$$

for some scalar parameter λ_i.

The advantage of using A-conjugate directions is that the convergence is quadratic, that is, if a quadratic function of n variables is being minimized, the minimum is reached in not more than n iterations. This can be proved as follows:

Consider the minimization by successive linear searches of the quadratic function

$$f(x) = f(h) + \frac{1}{2} (x - h)^T A (x - h) \tag{5}$$

for which the gradient is

$$g(x) = A (x - h) \tag{6}$$

By repeated use of equation (4), it is seen that

$$x_n = x_{j+1} + \sum_{i=j+1}^{n-1} \lambda_i \xi_i \tag{7}$$

for any j in $0 \leq j \leq n-1$. It then follows from equation (6), that

$$g_n = g_{j+1} + \sum_{i=j+1}^{n-1} \lambda_i A \xi_i \tag{8}$$

and therefore, using equation (3), that

$$g_n^T \xi_j = \sum_{i=j+1}^{n-1} \lambda_i \xi_i^T A \xi_j \tag{9}$$

If the vectors $\xi_0, \xi_1, \xi_2, \ldots \xi_{n-1}$ are A-conjugate, satisfying equation (2), then

$$g_n^T \xi_j = 0 \tag{10}$$

As the conjugate directions $\xi_0, \xi_1, \ldots \xi_{n-1}$ are linearly independent and form a basis,

$$g_n = 0$$

and hence, by equation (6),

$$x_n = h,$$

which is the required minimum.

Thus the minimum is located in the n^{th} iteration, or earlier if any particular λ_i equals zero, if the successive directions of search are A-conjugate.

As f(x) and g(x) are defined numerically, A is not explicitly available. The set of A-conjugate directions is generated in Fletcher and Reeves' method by a different procedure suggested by Hestenes and Stiefel [8]. In this method, successive conjugate directions, satisfying equation (2), are generated such that ξ_{i+1} is a linear combination of $-g_{i+1}$ and $\xi_0, \xi_1, \ldots \xi_i$, as follows:

$$\xi_{i+1} = -g_{i+1} + \lambda_i \xi_i \tag{11}$$

where

$$\lambda_i = \frac{g_{i+1}^2}{g_i^2} \tag{12}$$

A detailed proof that this method yields successive A-conjugate directions is given by Beckman [11].

Using this method of generating conjugate directions, Fletcher and Reeves established the following minimization algorithm:

$$
\begin{aligned}
x_0 &= \text{arbitrary} \\
g_0 &= g(x_0); \quad \xi_0 = -g_0 \\
x_{i+1} &= \text{position of minimum of } f(x) \text{ on the line through } x_i \text{ in} \\
&\qquad \text{the direction } \xi_i \qquad (13) \\
g_{i+1} &= g(x_{i+1}) \\
\lambda_i &= g_{i+1}^2 / g_i^2 \\
\xi_{i+1} &= -g_{i+1} + \lambda_i \xi_i
\end{aligned}
$$

This procedure is guaranteed, apart from rounding errors, to locate the minimum of any quadratic function of n variables in at most n iterations. For functions which are not quadratic, the process is iterative rather than n-step, and a test for convergence is required.

During early trials with this procedure, it was noticed that errors in the computation of the conjugate directions could accumulate and result in successive directions being nearly parallel [9]. This could result in slow convergence. Fletcher and Reeves modified the basic procedure so that periodically, all accumulated information about ξ_i was discarded, and the search restarted from the current x, in the direction of steepest descent, $-g_i$. The process remains quadratically convergent so long as such restarts are not more frequent than every n iterations. The modified procedure thus consists of (n+1) iterations.

The linear search problem in the iterative procedure (13) is basically a single variable search along each direction ξ_i. As in any practical application of the algorithm, the time spent to evaluate the function and the gradient may take up most of the computation time, it is preferable

to use a search procedure which reduces the number of these evaluations.

Fletcher and Reeves used a linear search procedure devised by Davidon [12] to find the minimum of the function along each direction. The procedure is described in detail in Appendix A.

The selection of a suitable convergence criterion is very important where the functions being minimized are not quadratic. If any g_i^2 vanishes, the iterations must stop. This is because it is the formal requirement for x_i to be at a minimum, and also to avoid division by zero. However, this condition is unlikely to be realized in practice, due to rounding off errors. A less stringent criterion would be to compare the value of g_i^2 with some tolerable assigned value. A second criterion is to compare the function values at the end of each cycle of (n+1) iterations and to terminate the search if the improvement is less than some small value. In some cases, it might be sufficient to continue the iterations until the change in each argument x_i is less than some assigned value. All these three criteria have been used in solving the problems in this report.

The minimization procedure (13) can be used to handle maximization problems by changing $-g_{i+1}$ into g_{i+1} and modifying the linear search procedure into one of maximization. Another method is to use the same procedure as for minimization and to consider the optimization problem as one of minimizing $-f(x)$. The latter method has been used to solve the inventory and advertising problem in this report.

3. THE CONJUGATE DIRECTION METHOD OF POWELL

There frequently arise optimization problems where it is difficult or practically impossible to calculate derivatives of the function. Powell [10] devised a method of searching along conjugate directions generated without the use of derivatives or the gradient vector. This method, like that of Fletcher and Reeves discussed earlier, has the advantage of quadratic convergence. It can be used for finding only the unconstrained minimum or maximum of a function.

The principle used in generating the conjugate directions is based on the fact that if x_0 is the minimum in a space containing the direction ξ, and x_1 is also the minimum in such a space, then the direction (x_1-x_0) is conjugate to ξ. The proof is as follows:

Consider the minimization of a quadratic function of n variables

$$f(x) = \overset{T}{x}Ax + Bx + C \tag{1}$$

If x_0 is the minimum along direction ξ,

$$\frac{\partial}{\partial\lambda}\{f(x_0 + \lambda\xi)\} = 0 \qquad \text{at } \lambda = 0. \tag{2}$$

Hence, substituting in equation (1), it is seen that

$$2\lambda\xi^T A\xi + 2Ax_0 + b\xi = 0, \quad \lambda = 0 \tag{3}$$

Also,
$$2\lambda\xi^T A\xi + 2Ax_1 + b\xi = 0, \quad \lambda = 0 \tag{4}$$

Hence
$$\xi^T A\,(x_1 - x_0) = 0 \tag{5}$$

which is the condition for conjugacy.

The convergence of a quadratic function of n variables to a minimum has already been proved to take place in n iterations, if the successive directions of search are A-conjugate.

Each iteration of Powell's procedure commences with a search down n linearly independent directions $\xi_1, \xi_2, \ldots \xi_n$, starting from the best known approximation to the minimum, x_0. These directions are chosen to be the coordinate directions initially, so that the start of the first iteration is identical to an iteration of the method of changing one variable at a time. In subsequent iterations, the method is modified to generate conjugate directions, by making each iteration define a new direction, ξ, and choosing the linearly independent directions for the next iteration to be $\xi_2, \xi_3, \ldots \xi_n, \xi$. An iteration of the basic procedure is hence as follows:

(i) For r = 1, 2, ... n calculate λ_r so that $f(x_{r-1} + \lambda_r \xi_r)$ is a minimum, and define $x_r = x_{r-1} + \lambda_r \xi_r$.

(ii) For r = 1, 2, ... n-1, replace ξ_r by ξ_{r+1}

(iii) Replace ξ_n by $(x_n - x_0)$

(iv) Choose λ so that $f\{x_n + \lambda(x_n - x_0)\}$ is a minimum and replace x_0 by $x_0 + \lambda(x_n - x_0)$

Powell found it necessary to modify the basic procedure in order to ensure that the rate of convergence to the minimum was satisfactory, even when the initial approximation was very poor. The reason was that on some occasions, the basic procedure may choose nearly dependent directions. The resultant directions may not hence span the full parameter space. Considering an instance, if say $\lambda_1 = 0$, then the modified procedure allows a direction other than ξ_1 to be discarded, so that the new set of directions will always be linearly independent. The modified procedure also allows for the search to be continued along the old set of directions in cases where it is unwise to replace any of them.

The procedure recommended by Powell is as follows:

(i) For r = 1, 2, ... n calculate λ_r so that $f(x_{r-1} + \lambda_r \xi_r)$ is a minimum, and define $x_r = x_{r-1} + \lambda_r \xi_r$.

(ii) Find the integer m, $1 \leq m \leq m$, so that $\{f(x_{m-1}) - f(x_m)\}$ is a maximum, and define $\Delta = f(x_{m-1}) - f(x_m)$

(iii) Calculate $f_3 = f(2x_n - x_0)$, and define $f_1 = f(x_0)$ and $f_2 = f(x_n)$.

(iv) If either $f_3 \geq f_1$ and/or

$$(f_1 - 2f_2 + f_3) \cdot (f_1 - f_2 - \Delta)^2 \geq \frac{1}{2} \Delta (f_1 - f_3)^2 \qquad (6)$$

use the old directions $\xi_1, \xi_2, \ldots \xi_n$ for the next iteration and use x_n for the next x_0, otherwise

(v) Define $\xi = (x_n - x_0)$ and calculate λ so that $f(x_n + \lambda\xi)$ is a minimum. Use $\xi_1, \xi_2, \ldots \xi_{m-1}, \xi_{m+1}, \ldots \xi_n, \xi$ as the directions and $x_n + \lambda\xi$ as the starting point for the next iteration.

The above modification is based on a very useful result. It is that, if the directions $\xi_1, \xi_2, \ldots \xi_n$ are scaled so that

$$\xi_i \, A \, \xi_i = 1, \quad i = 1, 2, \ldots n$$

the determinant of the matrix whose columns are the vectors ξ_i takes its maximum value if and only if the vectors are mutually conjugate. The proof of this result is given by Powell [10].

This result indicates that $\xi_1, \xi_2, \ldots \xi_n$ should be chosen to make the corresponding determinant as large as possible. The criterion is applied by using the new direction, ξ, defined by an iteration, if it

can cause the determinant to increase, and by rejecting the direction thereplacement of which causes the new determinant to be the largest. Powell has proved that the direction which should be discarded, if any, is ξ_m, $1 \leq m \leq n$, where m is such that $\{f(x_{m-1}) - f(x_m)\}$ is a maximum. The three function values f_1, f_2 and f_3 are used to predict the stationary value of the function along the new direction and to develop a criterion for the acceptance or rejection of the new direction $(x_n - x_0)$ for the next iteration. The reader is referred to Powell [10] for a detailed derivation of the criterion.

As the Powell's procedure is meant primarily for optimizing functions whose derivatives are not calculated, a linear search procedure for determining λ_r, not requiring the evaluation of derivatives, would be more suitable. The Fibonacci one dimensional search procedure has been chosen for this purpose, to solve the problems in this report. A detailed explanation of the method is given in Appendix A.

One convergence criterion used to terminate the search procedure is by comparing the function values at the end of every cycle and terminating if the improvement is less than an assigned value. This is the only criterion that has been used with this method for solving the problems in this report. However, other criteria could be devised like change in value of the function argument, depending on the requirements of the applications.

The minimization procedure (6) can easily be modified to solve maximization problems by making the following changes:

Modify the Fibonacci search procedure so as to maximize instead of minimize, in steps (i) and (v).

Find integer m, $1 \leq m \leq n$, so that $\{f(x_m) - f(x_{m-1})\}$ is a maximum, and define $\Delta = f(x_m) - f(x_{m-1})$, in step (ii).

Reverse the inequality signs in step (iv).

This modification has been used to solve the Inventory problem in the next section of this report.

4. APPLICATIONS

Two models representing production and inventory situations were considered.

The first model was a simple production planning situation involving one state variable. The second model was more complex and involved two state variables. Solutions were obtained by applying the Fletcher and Reeves' method and Powell's method for both models. A comparison of the two methods was made in each case.

All the problems were solved on an IBM 360 computer using computer programs coded in Fortran.

AN INVENTORY MODEL WITH ONE STATE VARIABLE

This model has been solved by Lee and Shaikh [1], using the functional gradient technique. Consider the case of a manufacturing facility where the rate of sales Q(t) is known with certainty. The rate of change of inventory level I(t) is given by

$$\frac{dI(t)}{dt} = P(t) - Q(t) \tag{1}$$

where P(t) = rate of production at time t.

The problem is to minimize the cost functional:

$$C_T = \int_0^T \left(C_I (I_m - I(t))^2 + C_p \exp (P_m - P(t))^2 \right) dt \tag{2}$$

where C_T is the total cost of inventory and production. C_P is the minimum production cost which occurs when the production rate equals P_m. The quantity P_m can be considered as the capacity of the manufacturing facility. An increase in production over this capacity may entail

additional equipment and manpower and can be very expensive. On the other hand, a decrease in production to a level below capacity can be equally expensive owing to the need for maintaining unused equipment and idle labor which cannot be reduced because of contract agreements. The quantity I_m can be considered as the storage capacity of inventory. C_I is the cost of carrying inventory. In many practical situations, the storage cost is a minimum when the storage capacity is completely utilized. Furthermore, the cost function given by equation (2) has a smoothing capability which is frequently desirable for many manufacturing processes. In this case, I_m and P_m can be considered as the desirable inventory and production levels, respectively. It is assumed that the sales forecast is known and given by the linear relation

$$Q(t) = a + bt \tag{3}$$

and the initial inventory is

$$I(0) = C \tag{4}$$

Equation (1), on substituting for Q(t), becomes

$$\frac{dI(t)}{dt} = P(t) - a - bt \tag{5}$$

In order to apply the conjugate gradient techniques to solve this problem, equations (2) and (5) are approximated by difference equations. From equation (5), the following difference equation is obtained:

$$x(t + \Delta t) = x(t) + (P(t) - a - bt)\ \Delta t \tag{6}$$

where $$x(t) = I(t) \tag{7}$$

To obtain the cost, the integral in equation (2) has to be computed over the limits $n\Delta t$ to $(n+1)\ \Delta t$. If t is small, this integral can be approximated by

$$\int_{n\Delta t}^{(n+1)\Delta t} \left[C_I(I_m - I(t))^2 + C_p \exp\ (P_m - P(t))^2\right] dt$$

$$= \left[C_I\ (I_m - I(t))^2 + C_p \exp\ (P_m - P(t))^2\right] \Delta t \qquad (8)$$

The following numerical values are assumed:

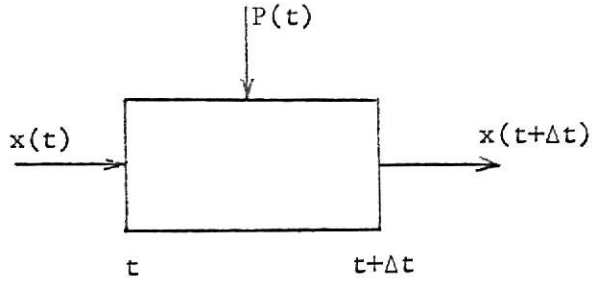
$a = 2$	$b = 1$	$c = 5$
$C_I = 0.1$	$I_m = 10$	$P_m = 5$
$C_p = 0.001$	$t_0 = 0$	$t_f = 1$

a. Solution by Fletcher and Reeves' method

The problem was first solved by applying the conjugate gradient method of Fletcher and Reeves. The number of stages into which the process should be divided is generally influenced by considerations of accuracy and computational cost. A large number of stages will give a better approximation to a continuous process and yield more accurate results. However, this results in increased computational time. For this study, solutions were obtained for a 5 stage and 10 stage process.

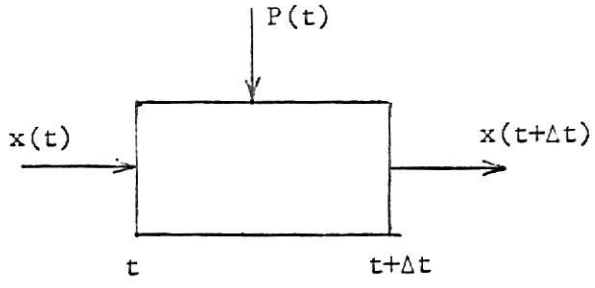
The computer program comprised of two subroutines. Subroutine FUNCT evaluated the inventory, function and gradient values required by the search subroutine FMCG. Subroutine FMCG was a standard subroutine from the IBM Scientific Subroutine Package and incorporated the Fletcher and Reeves' search procedure. This also included the one-dimensional search

Table 1. 5-stage process by Fletcher and Reeves' method



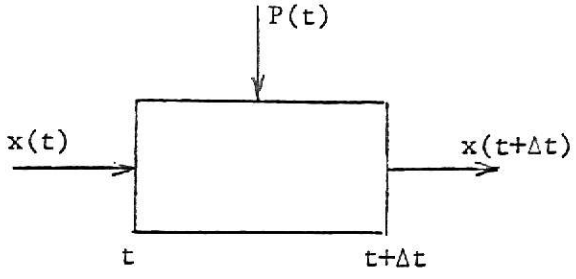
t	t+Δt	x(t)	P(t)	x(t+Δt)	C_T	Q(t+Δt)
0.0	0.2	5.00	7.15	5.99	0.43	2.20
0.2	0.4	5.99	7.09	6.93	0.69	2.40
0.4	0.6	6.93	6.89	7.79	0.84	2.60
0.6	0.8	7.79	6.64	8.55	0.91	2.80
0.8	1.0	8.55	6.78	9.31	0.94	3.00

Table 2. 10-stage process by Fletcher and Reeves' method



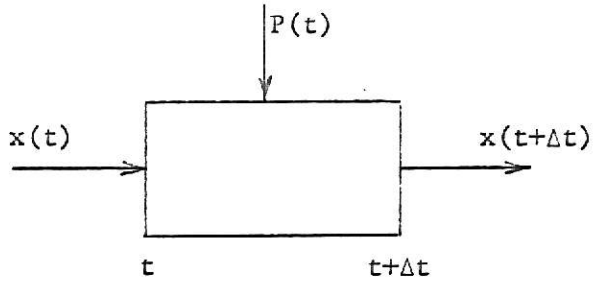
t	t+Δt	x(t)	P(t)	x(t+Δt)	C_T	Q(t+Δt)
0.0	0.1	5.00	7.17	5.51	0.24	2.10
0.1	0.2	5.51	7.13	6.00	0.43	2.20
0.2	0.3	6.00	7.08	6.48	0.58	2.30
0.3	0.4	6.48	7.03	6.94	0.69	2.40
0.4	0.5	6.94	6.98	7.39	0.77	2.50
0.5	0.6	7.39	6.90	7.82	0.84	2.60
0.6	0.7	7.82	6.77	8.23	0.88	2.70
0.7	0.8	8.23	6.62	8.61	0.90	2.80
0.8	0.9	8.61	6.49	8.97	0.92	2.90
0.9	1.0	8.97	6.63	9.33	0.93	3.00

Table 3. 5-stage process by Powell's method



t	t+Δt	x(t)	P(t)	x(t+Δt)	C_T	Q(t+Δt)
0.0	0.02	5.00	7.15	5.99	0.43	2.20
0.2	0.4	5.99	7.06	6.92	0.69	2.40
0.4	0.6	6.92	6.94	7.79	0.84	2.60
0.6	0.8	7.79	6.76	8.58	0.91	2.80
0.8	1.0	8.58	6.42	9.27	0.93	3.00

Table 4. 10-stage process by Powell's method



t	t+Δt	x(t)	P(t)	x(t+Δt)	C_T	Q(t+Δt)
0.0	0.1	5.00	7.17	5.51	0.24	2.10
0.1	0.2	5.51	7.13	6.00	0.43	2.20
0.2	0.3	6.00	7.08	6.48	0.57	2.30
0.3	0.4	6.48	7.03	6.94	0.69	2.40
0.4	0.5	6.94	6.96	7.39	0.77	2.50
0.5	0.6	7.39	6.89	7.82	0.84	2.60
0.6	0.7	7.82	6.80	8.23	0.88	2.70
0.7	0.8	8.23	6.68	8.61	0.90	2.80
0.8	0.9	8.61	6.51	8.97	0.92	2.90
0.9	1.0	8.97	6.15	9.29	0.93	3.00

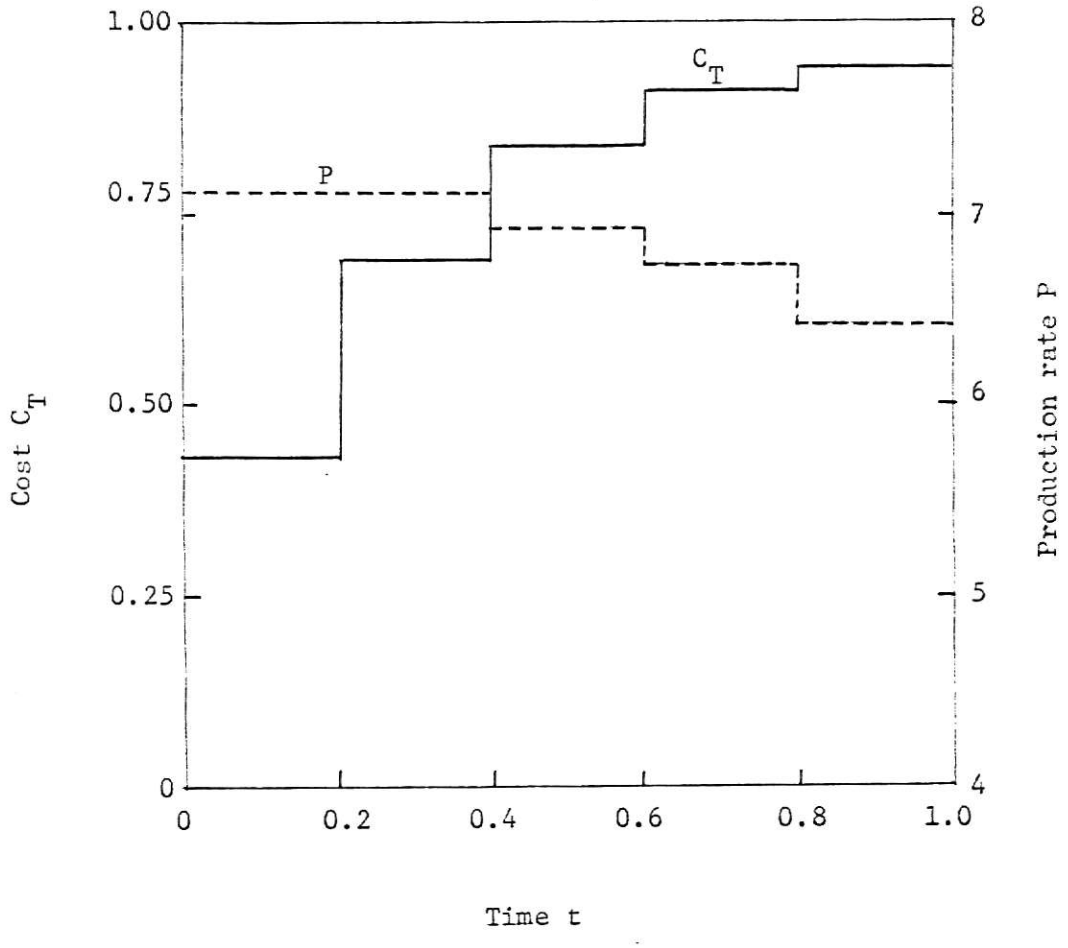


Fig. 1. Cost and production rate for the 5-stage process

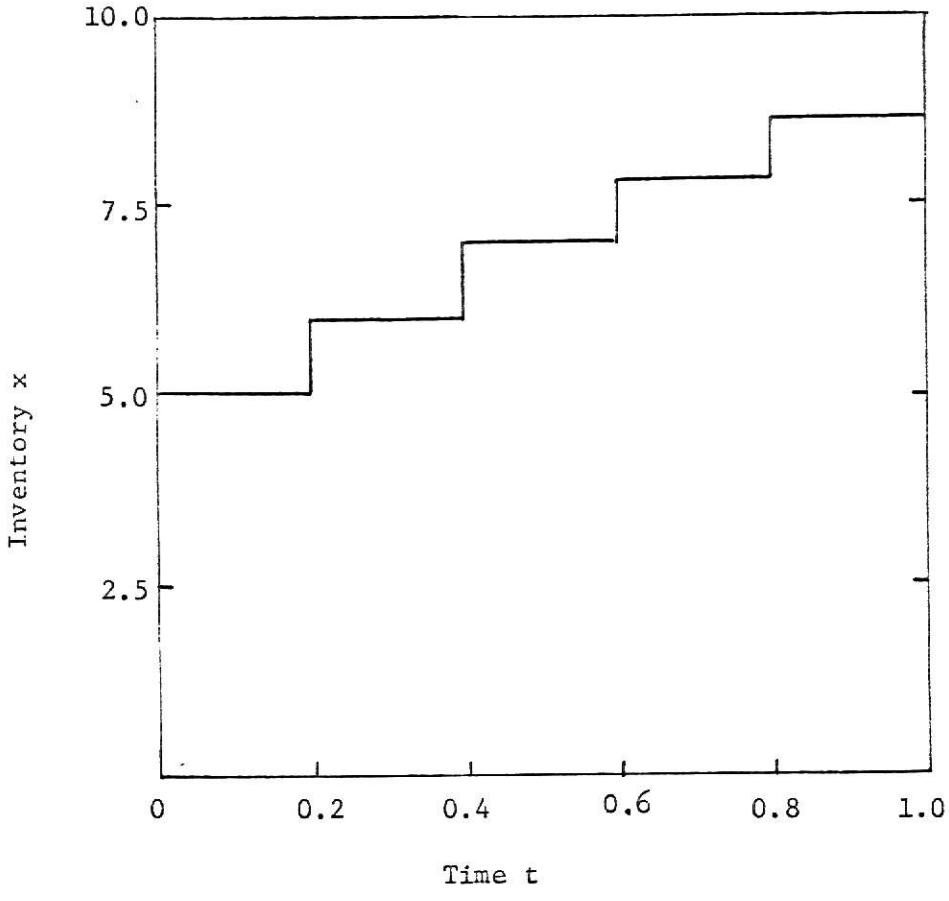


Fig. 2. Inventory for the 5-stage process

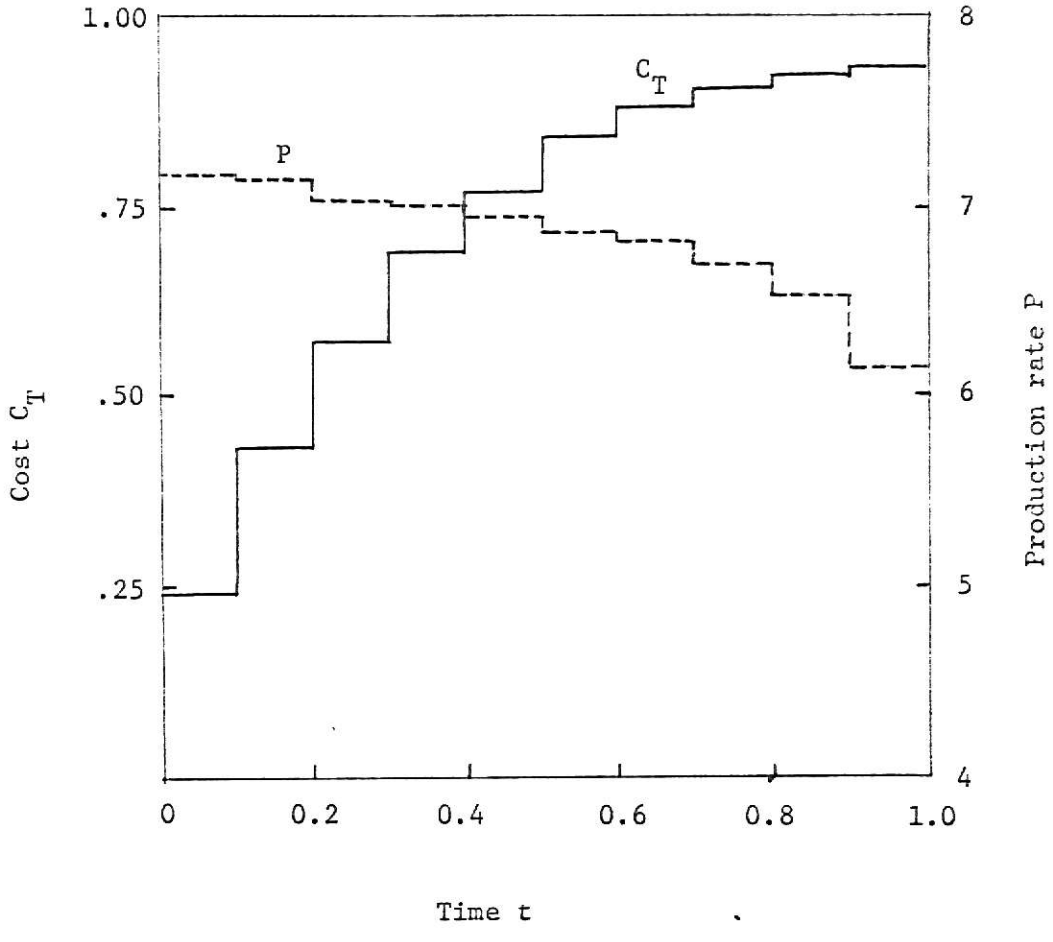


Fig. 3. Cost and production rate for the 10-stage process

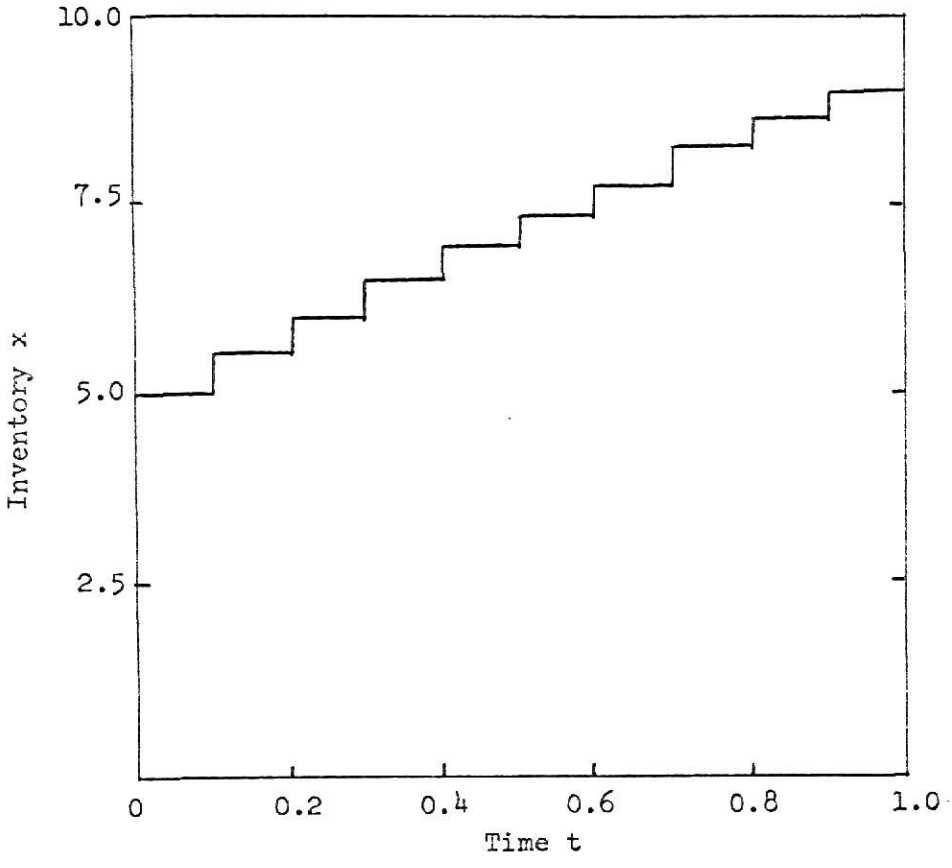


Fig. 4. Inventory for the 10-stage process

procedure of Davidon. The main program communicated the following parameters to the subroutines through the calling sequence:

(i) the number of variables, n,

(ii) initial approximation of the variables P_1, P_2, ... P_n

(iii) the required accuracy of computation, EPS,

(iv) an integer to limit the total number of iterations, LIMIT and

(v) an estimate of the value of the function at the unconstrained optimum, EST.

The flowchart for the computer program is given in Fig. 9, Appendix B and the program and sample results given in Appendix C.

Three different sets of values were used for the initial approximation to the variables P_1, P_2, ... P_n. It was observed that the optimal values remained the same irrespective of the initial approximation used. A summary of the results obtained is given in Tables 1 and 2. The optimal cost was 0.94 for a 5 stage process and 0.93 for a 10-stage process. There is no significant difference between the two values. It is therefore concluded that, for this particular model, the value of the optimal cost is not improved by increasing the number of stages. This is due to the fact that the inventory and cost functions are smooth functions, as can be seen from Figs. 1 and 2.

This problem has been solved by Lee and Shaikh [1] by using the functional gradient technique and the value of the optimal cost obtained by them was 0.94. There is thus excellent agreement with the results obtained with the Fletcher and Reeves' method.

b. Solution by Powell's method

The same problem was next solved by applying Powell's method. As

before, solutions were obtained for a 5 stage and a 10 stage process.

The computer program comprised of three subroutines. Subroutine POWELL incorporated Powell's search procedure. Subroutine FIBON calculated the unconstrained optimum along each direction computed by Powell's method, using the Fibonacci search procedure. Subroutine FUNCT evaluated the inventory and function values required by the main subroutine POWELL. The main program communicated the following parameters to the subroutines through the calling sequence:

(i) the number of variables, n,

(ii) initial approximation of the variables P_1, P_2, ... P_n,

(iii) the required accuracy of computation, EPS, and

(iv) an integer to limit the total number of iterations, LIMIT.

The flowchart for the computer program is given in Fig. 11, Appendix B and the program and sample results given in Appendix C.

As in the case of Fletcher and Reeves' method, the problem was solved using three different sets of values as the initial approximation to the variables P_1, P_2, ... P_n. Identical results were obtained in each case. A summary of the results is given in Tables 3 and 4. The value of the optimal cost was seen to be 0.93 for both the 5 and 10 stage processes. This confirms the earlier conclusion that, in this problem, an increase in the number of stages does not yield a more accurate value of the optimal cost.

c. Comparison

A comparison of the results obtained by the two methods gave similar values. However, Powell's method required about 35% to 130% more computing time. This was because Powell's procedure necessitated a large number

Table 5. Convergence rates for the 5 stage process

No. of Iterations	Fletcher & Reeves' method				Powell's method			
	P_1	P_3	P_5	C_T	P_1	P_3	P_5	C_T
	Initial approximation: All P_i = 1.0							
0	1.00	1.00	1.00	8889.39	1.00	1.00	1.00	8889.39
1	6.44	6.44	6.44	1.03	5.96	5.98	5.99	1.14
2	7.21	6.73	6.64	0.94	7.18	6.95	6.42	0.93
3	7.17	6.83	6.72	0.94	7.15	6.94	6.42	0.93
4	7.15	6.84	6.78	0.94				
5	7.15	6.89	6.78	0.94				
6	7.15	6.89	6.78	0.94				
	Initial approximation: All P_i = 5.0							
0	5.00	5.00	5.00	1.43	5.00	5.00	5.00	1.43
1	7.29	6.11	5.66	1.00	7.20	7.10	6.41	0.94
2	7.13	6.45	5.89	0.96	7.15	7.06	6.42	0.93
3	7.20	6.98	6.32	0.94				
4	7.16	7.03	6.42	0.94				
5	7.14	7.02	6.54	0.94				
7	7.17	6.96	6.74	0.94				
10	7.16	6.95	6.78	0.94				
11	7.16	6.95	6.78	0.94				
	Initial approximation: All P_i = 10.0							
0	10.00	10.00	10.00	****	10.00	10.00	10.00	****
1	6.99	6.99	6.99	0.95	5.00	5.00	5.00	1.41
2	7.20	6.95	6.93	0.94	7.20	6.96	6.41	0.94
3	7.11	6.93	6.88	0.94	7.15	6.94	6.42	0.93
5	7.16	6.93	6.86	0.94				
6	7.16	6.93	6.86	0.94				

Table 6. Convergence rates for the 10 stage process

No. of Iterations	Fletcher & Reeves' method				Powell's method			
	P_1	P_5	P_{10}	C_T	P_1	P_5	P_{10}	C_T
	Initial approximation: All $P_i = 1.0$							
0	1.00	1.00	1.00	8889.36	1.00	1.00	1.00	8889.36
1	6.44	6.44	6.44	1.02	5.85	5.90	5.99	1.16
2	7.25	6.76	6.51	0.93	7.21	6.98	6.14	0.93
3	7.15	6.87	6.54	0.93	7.17	6.96	6.15	0.93
4	7.16	6.95	6.57	0.93				
10	7.18	6.98	6.63	0.93				
11	7.17	6.98	6.63	0.93				
12	7.17	6.98	6.63	0.93				
	Initial approximation: All $P_i = 5.0$							
0	5.00	5.00	5.00	1.41	5.00	5.00	5.00	1.41
1	7.62	6.31	5.34	1.04	7.23	7.01	6.13	0.93
2	7.62	6.31	5.34	1.04	7.17	6.97	6.15	0.93
3	7.62	6.33	5.35	1.04				
4	7.52	6.47	5.38	1.01				
10	7.23	7.06	5.65	0.94				
15	7.22	6.92	6.17	0.93				
20	7.18	7.00	6.55	0.93				
21	7.19	6.99	6.59	0.93				
22	7.19	6.99	6.59	0.93				
	Initial approximation: All $P_i = 10.0$							
0	10.00	10.00	10.00	****	10.00	10.00	10.00	****
1	6.99	6.99	6.99	0.94	5.00	5.00	5.00	1.41
2	7.22	6.97	6.87	0.93	7.23	7.01	6.13	0.93
3	7.11	6.96	6.76	0.93	7.17	6.96	6.15	0.93
4	7.20	6.96	6.70	0.93				
10	7.18	6.97	6.69	0.93				
11	7.18	6.97	6.69	0.93				

of function evaluations during every iteration. As function evaluations took up most of the computing time in both the methods, this made a significant difference to their computing times.

In comparing the convergence rates for the two methods, it was found that Powell's method required a fewer number of iterations for convergence. Tables 5 and 6 gives the convergence rates for each method for 5 and 10 stage processes. It is evident that Powell's method is definitely faster in converging to the optimum, though it takes more computational time.

AN INVENTORY AND ADVERTISING MODEL WITH TWO STATE VARIABLES

This model is an extension of an advertising model formulated by Teichroew [14]. Consider a marketing situation where only a certain number of potential customers possess information about a company's product. Assume that the total number of persons in the group under consideration remains constant and that diffusion of information occurs only through personal contact. The number of "contacts" made by an average informed person in an arbitrary unit of time is given by a "contact coefficient". This coefficient is a fixed number which is the same for all members of the group. In a contact, the contactee receives the information if he does not already have it; if he already has it, the contact is wasted so far as increasing the number of people who have the information is concerned.

Let K(0) = number of informed persons at time 0

K(t) = number of informed persons at time t

N = total number of persons in the group

C = contact coefficient i.e the number of contacts made by one informed person per unit time

Then $K(t)/N$ = proportion of informed persons in the group at time t.

$1-K(t)/N$ = proportion of uninformed persons at time t.

$CK(t)\cdot dt$ = contacts made during a time interval dt.

The increase in total number of informed persons, $dK(t)$, during a short interval of time dt is obtained only from contacts made with uninformed persons.

Thus $$dK(t) = CK(t)\ dt \left(1 - \frac{K(t)}{N}\right)$$

This is the differential equation:

$$\frac{dK(t)}{dt} = CK(t) \left(1 - \frac{K(t)}{N}\right) \tag{1}$$

Suppose that the company can increase the number of contacts by spending money on advertising. In particular, it can increase the number of contacts made by the informed persons, by an additional number A per unit of time. Eqn. (1) then becomes

$$\frac{dK(t)}{dt} = K(t)(C + A(t)) \left(1 - \frac{K(t)}{N}\right) \tag{2}$$

If each successful contact results in the sale of n units of the company's product and if $Q(t)$ represents the sale at time t, then

$$Q(t) = nK(t)$$

Assuming $n = 1$ for a particular product and substituting $Q(t)$ for $K(t)$ in eqn (2), we get

$$\frac{dQ(t)}{dt} = Q(t)(C + A(t)) \left(1 - \frac{Q(t)}{N}\right) \tag{3}$$

Next, the rate of change of the company's inventory is given by

$$\frac{dx(t)}{dt} = P(t) - Q(t) \tag{4}$$

where $P(t)$ = production rate at time t

The production rate for the product under consideration is assumed to be a linear function given by

$$P(t) = a + bt \tag{5}$$

where a and b are known constants. This assumption simplifies the model by avoiding a second control variable. The company's objective is to maximize the profit

$$S_T = \int_0^T \left(FQ(t) - C_I(P_I - x(t)^2 - C_A\, AQ(t)\right) dt \tag{6}$$

where S_T is the total net profit, F is the revenue from the sale of one unit of the product, C_I is the inventory carrying cost and C_A is the cost of advertising. P_I may be considered as the available storage capacity for the product.

Equations (3) through (6) completely represent the system. It has two state variables, x(t) and Q(t) and one control variable, A(t). In order to apply the conjugate gradient techniques to solve this model, the differential equations are approximated by difference equations.

Thus, equation (3) is reduced to

$$Q(t + \Delta t) = Q(t) + Q(t)\left[C + A(t)\right]\left[1 - \frac{Q(t)}{N}\right]\Delta t \tag{7}$$

If this equation is used, it is possible for $Q(t + \Delta t)$ to exceed N, which

is not possible in the physical system being considered. To overcome this difficulty, the term $(1 - Q(t)/N)$ is replaced by $(1 - Q(t + \Delta t)/N)$ and this makes it impossible for $Q(t + \Delta t)$ to exceed N. By incorporating this change and rearranging the terms equation (7) becomes

$$Q(t + \Delta t) = \frac{Q(t)\left(1 + (C + A(t)\ t\right)}{1 + (C + A(t))\ Q(t)\Delta t/N} \tag{8}$$

Equation (4) is reduced to

$$x(t + \Delta t) = x(t) + (P(t) - Q(t))\ \Delta t \tag{9}$$

To obtain the profit, the integral in equation (6) has to be computed between limits $n\Delta t$ and $(n+1)\Delta t$. If Δt is small, the integral can be approximated by

$$S_T = \left(FQ(t) - C_I(P_I - x(t)^2 - C_A\ AQ(t)\right)\ \Delta t \tag{10}$$

The initial conditions and numerical values for the constants used are:

$P_I = 50$	$b = 100$	$x(0) = 20$
$C_I = 0.15$	$a = 70$	$Q(0) = 20$
$C_A = 1.5$	$c = 2$	$T = 1$
$N = 150$	$F = 10$	

It is assumed that the company has unlimited funds for advertising and thereby no constraint has to be imposed on the control variable A(t).

a) Solution by Fletcher and Reeves' method

An attempt was made to solve this problem for 5 and 10 stage processes. The computer program was similar to the one used for solving the Inventory

problem and comprised of two subroutines FUNCT and FMCG. As this problem was one of maximization, subroutine FUNCT was designed to yield the negative of the function and gradient values to the main subroutine FMCG. It was thus possible to use the original IBM subroutine FMCG, which was designed as a minimization procedure, without making any changes. Subroutine FUNCT evaluated the values of inventory, production and sales, apart from function and gradient values required by the main subroutine. The main program communicated the initial approximation to the variables A_1, A_2, ... A_n to the subroutines in addition to the parameters n, EPS and LIMIT, through the calling sequence.

It was noticed that in the very first iteration, the control variables picked up negative values. This could not be avoided because the Fletcher and Reeves' method is not designed to recognize even non-negativity constraints on the control variables. As negative values were not permissible in this problem, the control variable being the amount of advertising in each stage, the problem could not be solved by this method. Further, it can be seen that the function represented by equation (10) is non-decreasing for all negative values of the control variable, and having an infinite maximum. This caused the failure of the one-dimensional search procedure used in the subroutine FMCG and explains why the iterations did not continue in the negative region.

The computer program for this problem and sample results are given in Appendix C.

b) Solution by Powell's method

The problem was solved for both 5 stage and 10 stage processes. The computer program conprised of three subroutines FUNCT, FIBON and POWELL. The main subroutine POWELL was modified into a maximization method by

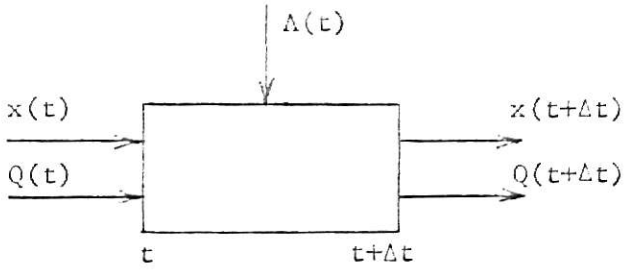
making the necessary changes in the iterative procedure. The one dimensional search procedure was also changed to one of maximization by modifying subroutine FIBON. Subroutine FUNCT was changed to evaluate the sales, production, inventory and function values for this problem. The main program communicated the initial approximation to the variables A_1, A_2, ... A_n, to the subroutines, in addition to the parameters n, EPS and LIMIT, through the calling sequence.

The problem was solved using two sets of values as the initial approximation to the variables A_1, A_2, ... A_n. Identical results were obtained in each case.

Powell's method, like the Fletcher and Reeves' method, was not designed to recognize non-negativity constraints on the control variables. However, it was possible in Powell's method to adjust the rate at which the control variable progressed into the negative region. This was done by setting suitable limits on the interval of interest in the Fibonacci search procedure, which slowed down any decrease in value of the control variables without imposing any restriction on its increase. Thus it was possible to choose the optimal value satisfying the non-negativity requirements of the problem, even though the iterations continued into the negative region.

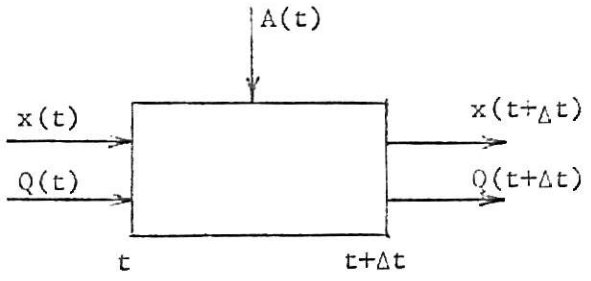
The results are given in Tables 7 and 8. The values of the optimal profit were 607.62 for the 5 stage process and 683.50 for the 10 stage process. Thus a significant improvement was noticed in the case of the 10 stage process. This was because a 5 stage process with two state variables is a very approximate representation of the real situation. The profit increased from 607.62 to 683.50, an increase of 12.5%. Sales rate

Table 7. 5-stage process by Powell's method



t	t+Δt	x(t)	Q(t)	A(t)	x(t+Δt)	Q(t+Δt)	P(t)	S
0	0.2	20.00	20.00	7.87	28.58	47.10	70.00	-30.82
0.2	0.4	28.58	47.10	2.96	36.27	71.53	90.00	43.10
0.4	0.6	36.27	71.53	0.34	45.11	85.82	110.00	205.39
0.6	0.8	45.11	85.82	0	55.55	97.78	130.00	400.02
0.8	1.0	55.55	97.78	0	67.84	108.58	150.00	607.62

Table 8. 10-stage process by Powell's method



t	t+Δt	x(t)	Q(t)	A(t)	x(t+Δt)	Q(t+Δt)	P(t)	S
0	0.1	20.00	20.00	12.49	23.90	41.05	70.00	-46.05
0.1	0.2	23.90	41.05	6.14	26.81	60.89	80.00	-49.28
0.2	0.3	26.81	60.89	2.56	29.33	74.80	90.00	- 9.59
0.3	0.4	29.33	74.80	0.53	32.01	83.21	100.00	62.19
0.4	0.5	32.01	83.21	0	35.02	89.88	110.00	148.71
0.5	0.6	35.02	89.88	0	38.39	96.32	120.00	243.00
0.6	0.7	38.39	96.32	0	42.14	102.43	130.00	344.50
0.7	0.8	42.14	102.43	0	46.30	108.14	140.00	452.44
0.8	0.9	46.33	108.14	0	50.99	113.42	150.00	565.84
0.9	1.0	50.99	113.42	0	56.17	118.22	160.00	683.50

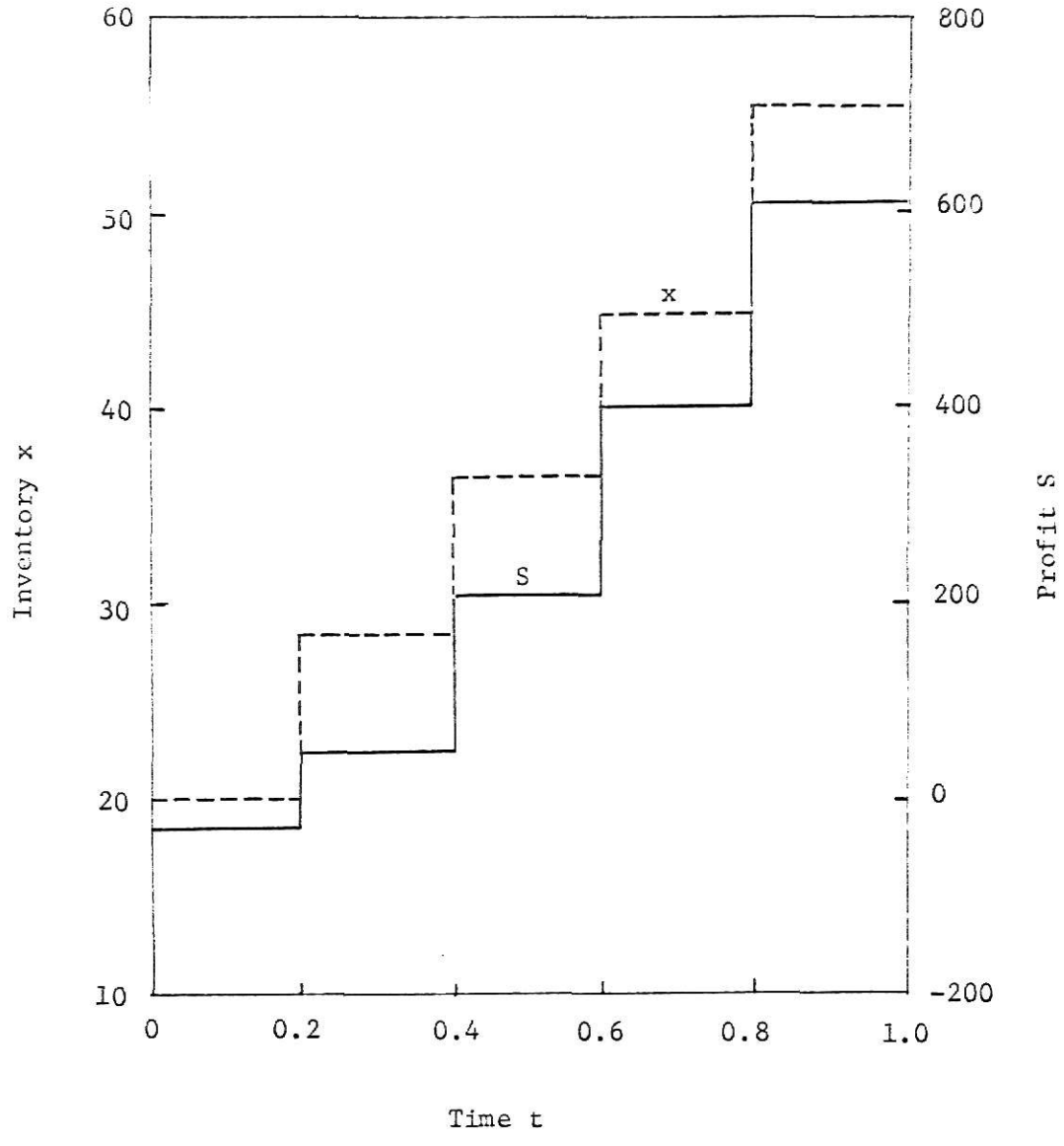


Fig. 5. Inventory and profit for the 5-stage process

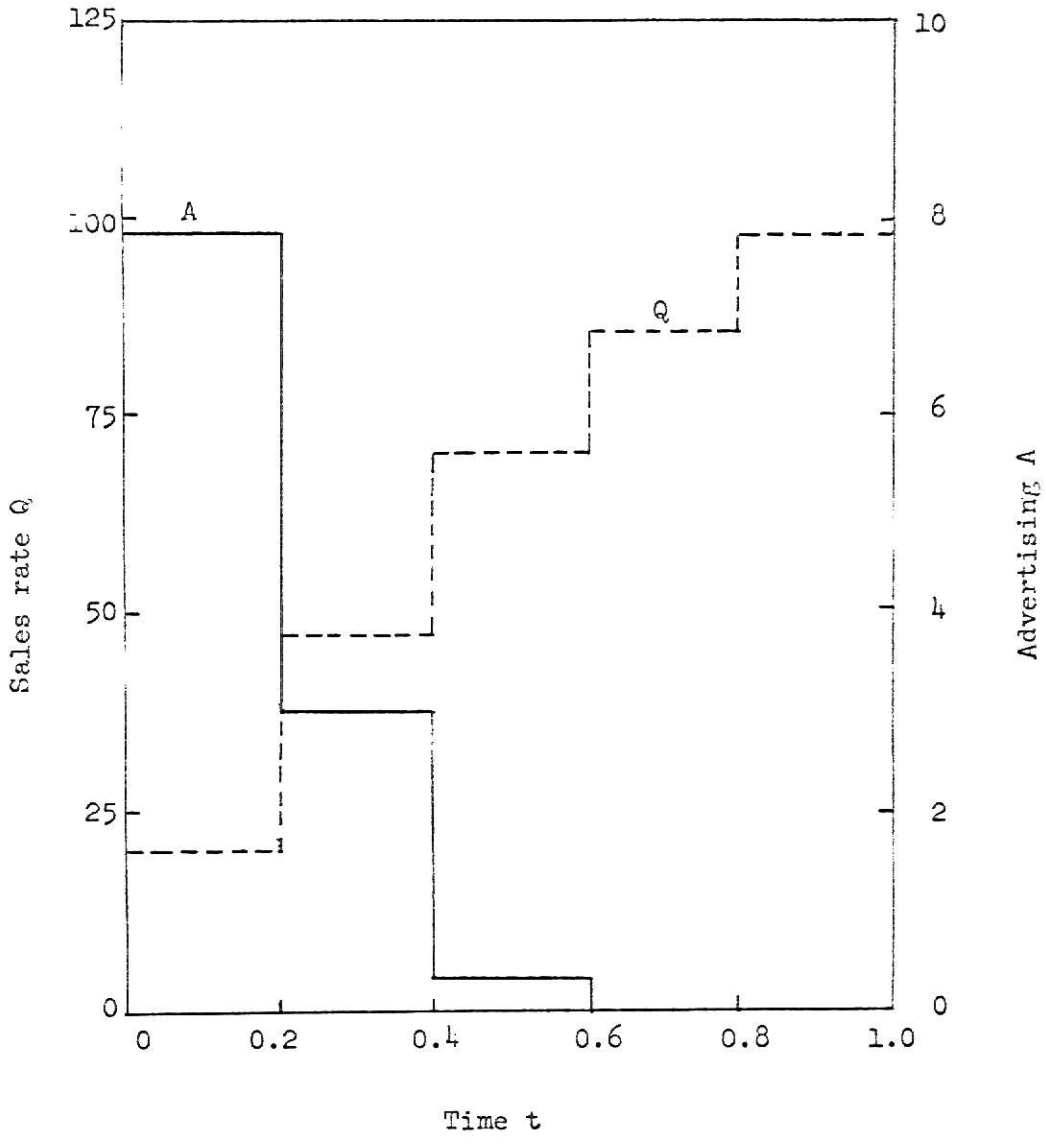


Fig. 6. Sales rate and advertising for the 5-stage process

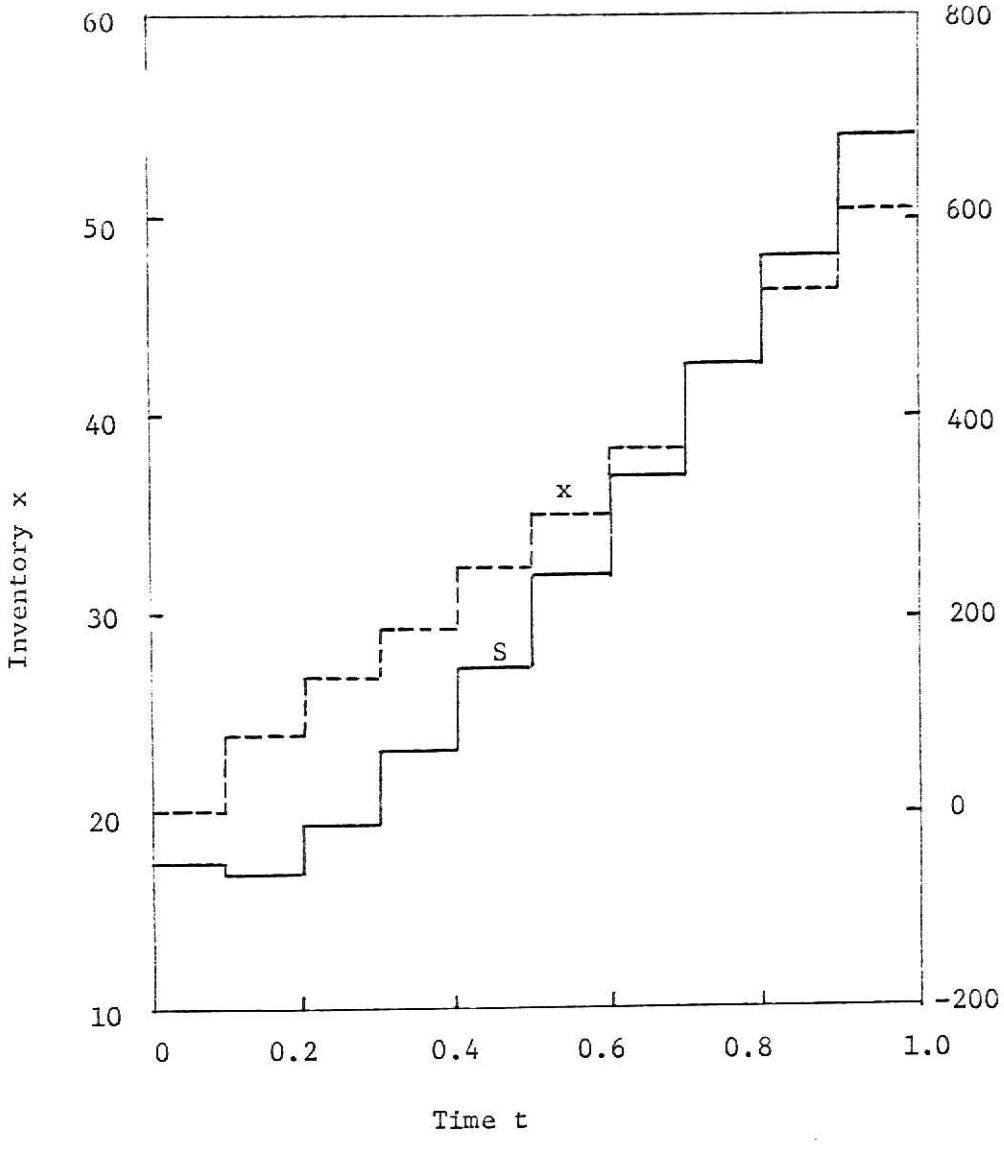


Fig. 7. Inventory and profit for the 10-stage process

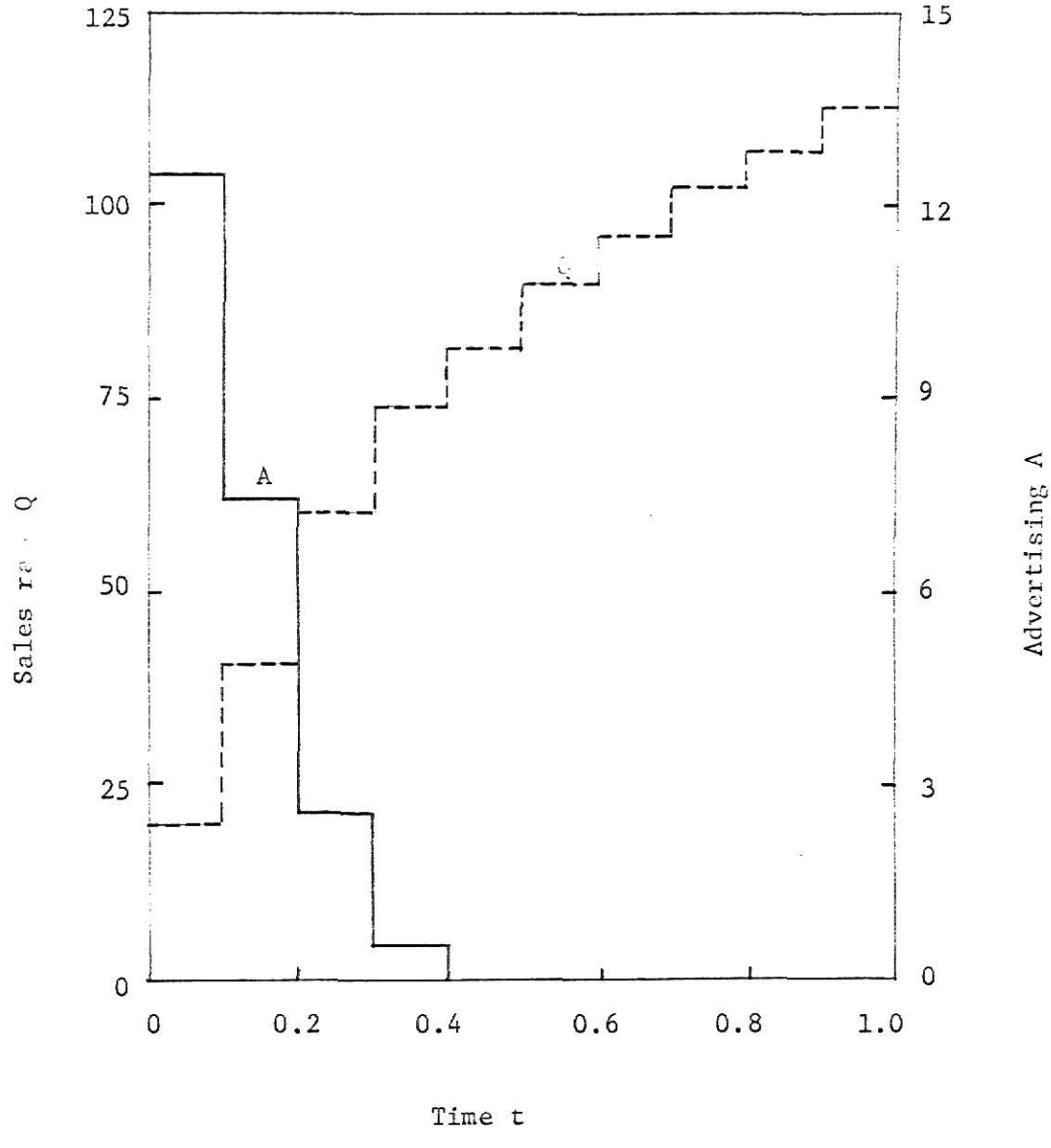


Fig. 8. Sales rate and advertising for the 10-stage process

increased from 108.58 to 118.22, an increase of 9.2%. This was because in a 10 stage process, the advertising expenditure was utilized better, as the effect of advertising was felt sooner, than in the case of a 5 stage process.

The computer program and sample results for this problem are given in Appendix C.

c) Comparison

As no solution was obtained by the Fletcher and Reeves' method for this problem, a comparison of the effectiveness of the two methods could not be made. However, in the light of the experience gained in trying to solve this problem, a few conclusions could be drawn. The calculation of the gradients in the case of Fletcher and Reeves' method was a tedious affair, as both inventory and sales were functions of the control variable. Where optimization problems involving a larger number of state variables are being considered, this might prove to be a major drawback with the method. Powell's method, on the other hand, does not have any such limitation.

Table 9 gives the convergence rates for 5 and 10 stage processes by Powell's method. This problem has again shown the ability of the method to converge rapidly to the optimum.

Table 9. Convergence rates for the 5 and 10 stage processes by Powell's method.

No of Iterations	5-stage process				10-stage process			
	A_1	A_3	A_5	S	A_1	A_5	A_{10}	S
	Initial approximation: All $A_i = 5.0$							
0	5.00	5.00	5.00	219.95	5.00	5.00	5.00	217.97
1	4.00	4.00	4.00	337.65	4.00	4.00	4.00	349.57
2	3.82	3.00	3.00	427.22	4.04	3.00	3.00	450.69
3	5.29	2.00	2.00	502.46	7.47	2.00	2.00	543.90
4	6.87	1.00	1.00	564.29	10.42	1.00	1.00	621.92
5	7.87	0.34	0	607.62	12.49	0	0	683.50
	Initial approximation: All $A_i = 10.0$							
0	10.00	10.00	10.00	-619.60	10.00	10.00	10.00	-681.38
1	9.00	9.00	9.00	-432.54	9.00	9.00	9.00	-484.35
2	8.00	8.00	8.00	-252.23	8.00	8.00	8.00	-293.08
3	7.00	7.00	7.00	- 80.94	7.00	7.00	7.00	-109.77
4	6.00	6.00	6.00	78.05	6.00	6.00	6.00	62.27
5	5.00	5.00	5.00	219.95	5.00	5.00	5.00	217.97
6	4.00	4.00	4.00	337.65	4.00	4.00	4.00	349.57
7	3.82	3.00	3.00	427.22	4.04	3.00	3.00	450.69
8	5.29	2.00	2.00	502.46	7.47	2.00	2.00	543.90
9	6.87	1.00	1.00	564.29	10.42	1.00	1.00	621.92
10	7.87	0.34	0	607.62	12.49	0	0	683.50

5. DISCUSSION

Any effective comparison of direct methods of optimization like those studied in this report is faced with several difficulties. In the first place, the relative performance of the methods is bound to differ with the problems chosen. Secondly, while one method may terminate with a fewer number of iterations, another may come out with a more accurate result and take more iterations to converge to the optimum. Leon [16] and Rosenbrock and Storey [17] have suggested the use of accuracy, number of function evaluations and computing time as a basis of comparison. The need for considering the number of function evaluations arises because the most difficult problems to handlein practice are those where the function evaluation is very slow [17]. For example, the function may be determined by the numerical solution of a set of differential equations. The computing time will then depend much less on the logic of the search and much more on the number of evaluations.

The accuracy of the results obtained by both the methods is nearly the same. No attempt was made to count the total number of function evaluations made by the two methods for each problem. However, it is obvious from the computer programs that Powell's method requires more function evaluations. In addition, Fletcher and Reeves' have used Davidon's linear search procedure which is designed to reduce the number of function evaluations by doubling the step size each time. In Powell's method, the Fibonacci method is used, which results in a fixed number of function evaluations being made every time the subroutine is called. Davidon's search procedure requires the evaluation of derivatives. Hence, it was not considered advisable to incorporate it in Powell's method

which is meant to be used primarily in cases where derivatives are difficult or impossible to calculate.

On the basis of both computing time and number of function evaluations, Fletcher and Reeves' method appears to be superior to Powell's method. However, by requiring fewer number of iterations for convergence, Powell's method has shown itself to be quite a powerful method of optimization. It has the added advantage of not requiring any derivatives to be calculated. Even in the relatively simple two state variable problem, the determination of the gradient vector, in Fletcher and Reeves' method, involved tedious calculations.

The main drawback with both the methods is their inability to recognize constraints on variables. This limitation restricts their application to a very small class of problems. Even in the case of a management problem involving constraints, useful information can be gained from a study of the results obtained for the unconstrained problem. In such cases, the two methods appear to be quite suitable for use.

The problems solved by the two methods show that both the methods have the property of quadratic convergence, even with poor approximations to the optimum as starting values. Thus they represent a significant improvement over the earlier gradient methods.

ACKNOWLEDGEMENT

The author wishes to express his deep sense of gratitude to his major professor, Dr. E. S. Lee, for his guidance, constructive criticism, kindness and understanding and the personal interest he has taken in the preparation of this master's report.

REFERENCES

1. Lee, E. S. and Shaikh, M. A., "Optimal Production Planning by a Gradient Technique - I. First Variations," an unpublished paper, Kansas State University, Manhattan, Kansas.

2. Rosenbrock, H. H., "An automatic method for finding the greatest or the least value of a function," The Computer Journal, Vol. 3, No. 3, 1960, pp. 175 - 184.

3. Hooke, R. and Jeeves, T. A., "Direct Search Solution of Numerical and Statistical Problems," Journal of Assoc. Computing Mach., Vol. 8, No. 2, 1961, pp. 212 -229.

4. Lee, E. S., "Search Techniques," Lecture notes, Kansas State University, Manhattan, Kansas, 1967.

5. Nelder, J. A. and Mead, R., "A Simplex method for function minimization," The Computer Journal, Vol. 7, No. 4, 1965, pp. 308 - 313.

6. Hadley, G., Non-linear and Dynamic Programming, Addison-Wesley, Reading, 1964.

7. Wilde, D. J., Optimum Seeking Methods, Prentice-Hall, New Jersey, 1964.

8. Hestenes, M. R. and Stiefel, E., "Methods of conjugate gradients for solving linear systems," Journal of Research, N.B.S., Vol. 49, No. 6, 1952, pp. 409 - 436.

9. Fletcher, R. and Reeves, C. M., "Function minimization by conjugate gradients," The Computer Journal, Vol. 7, No. 2, 1964, pp. 149 - 154.

10. Powell, M. J. D., "An efficient method for finding the minimum of a function of several variables without calculating derivatives," The Computer Journal, Vol. 7, No. 2, 1964, pp. 155 - 162.

11. Beckman, F. S., "The solution of linear equations by the conjugate gradient method," in Mathematical Methods for Digital Computers, Ralston, A. and Wilf, H. S. (Eds.), John Wiley, New York, 1960.

12. Davidon, W. C., "Variable metric method for minimization," A.E.C. Research and Development Report, ANL-5990 (Rev.), 1959.

13. Gue, R. L. and Thomas, M.E., Mathematical Methods in Operations Research, The Macmillan Company, New York, 1968.

14. Teichroew, D., An Introduction to Management Science - Deterministic models, John Wiley, New York, 1966.

15. Fletcher, R., "Function minimization without evaluating derivatives - a review," The Computer Journal, Vol. 8, No. 1, 1965, pp. 33 - 41.

16. Leon, A., "A comparison among eight known optimizing procedures" in Recent advances in Optimization, Lavi, A. and Vogl, T. P. (Eds.), John Wiley, New York, 1966.

17. Rosenbrock, H. H. and Storey, C., Computational Techniques for Chemical Engineers, Pergamon Press, Oxford, 1966.

APPENDIX A

DAVIDON'S LINEAR SEARCH PROCEDURE USED IN FLETCHER & REEVES' METHOD OF OPTIMIZATION

Fletcher and Reeves have used the method adopted by Davidon in his variable metric method for minimization [12]. The conjugate gradient method requires the position of the maximum or minimum of the function $f(x)$ to be found along a line passing through the point x_i in the direction ξ_i, where x_i and ξ_i are computed in the main program. Essentially, the linear search problem is to determine λ such that

$$y'(\lambda_m) = 0 \tag{1}$$

where

$$y(\lambda) = f(x_i + \lambda\xi_i) \tag{2}$$

and therefore

$$y'(\lambda) = \xi_i^T g\ (x_i + \lambda\xi_i) \tag{3}$$

Thus $y(\lambda)$ and $y'(\lambda)$ can be calculated for any λ and, in particular, the values $y(0)$ and $y'(0)$ at the starting points are already known.

The calculation is in three stages; the first estimates the order of magnitude of λ_m, the second establishes bounds on it, and the third interpolates its value. The step size, h, for λ is arbitrarily chosen by Fletcher and Reeves to be equal to the displacement along ξ_i of unit length in the x-space. This is supplemented with the requirement that there be available an estimate, est, of the value of the function $f(x)$ at the unconstrained maximum or minimum. On the assumptions that this is a correct estimate, that the unconstrained maximum or minimum lies on the line $x_i + \lambda\xi_i$, and that $f(x)$ is quadratic, the value k for λ_m is given by

$$k = \lambda_m = 2(est - f_i)/\xi_i^T g_i \tag{4}$$

As the unconstrained maximum or minimum will generally not lie on the line, equation (4) tends to over-estimate λ_m. This leads to the following criterion in determining the step size:

$$h = k \quad \text{if} \quad 0 < k < (\xi_i^2)^{-1/2}$$

$$= (\xi_i^2)^{-1/2} \qquad \text{otherwise.}$$

In the second stage y' is examined at the points $\lambda = 0, h, 2h, 4h, \ldots a, b$, where λ is doubled each time, and where b is the first of these values at which y' is non positive or nonnegative, depending on whether the linear search is one of maximization or minimization respectively. It then follows that λ_m is bounded in the interval $a < \lambda_m \leq b$.

The third stage uses cubic interpolation to determine an estimate λ_e of the stationary point λ_m. Assuming that the function f(x) is a cubic equation of the form $f(x) = Ax^3 + Bx^2 + Cx + D$, and from a knowledge of the values of the function and derivative at a and b, it can be easily shown [4] that

$$\lambda_e = b - \left(\frac{y'(b) + w - z}{y'(b) - y(a) + 2w} \right)(b - a) \tag{5}$$

where z and w are defined as follows:

$$z = \frac{3y(a) - y(b)}{b-a} + y'(a) + y'(b) \tag{6}$$

$$w = \pm (z^2 - y'(a)y'(b))^{1/2} \tag{7}$$

The positive value of w is used if a minimum is being located and the negative value in case of a maximum.

For a minimization problem, if neither y(a) nor y(b) is less than $y(\lambda_e)$, then λ_e is accepted as the estimate of λ_m. Otherwise, according as $y'(\lambda_e)$ is positive or negative, the interpolation is repeated over the sub-interval (a,λ_e) or (λ_e, b), respectively.

A similar reasoning is used to repeat the interpolation over a reduced interval in the case of a maximization problem.

The computer flowchart for the search procedure is given in Appendix B. The program has been coded as part of the Fletcher and Reeves' search subroutine and is given in Appendix C.

The main advantage of this search technique is that very few function and gradient evaluations are required to locate the stationary point. This is because the step size is doubled each time during the search. Most of the computational time required in gradient techniques is taken up in function evaluations. Thus a reduction in these evaluations will significantly reduce computation time.

THE FIBONACCI SEARCH PROCEDURE USED IN POWELL'S METHOD OF OPTIMIZATION

The first step in the iterative procedure of Powell's method is to determine, for r = 1, 2, ... n the value of λ_r which maximizes or minimizes the function $f(p_{r-1} + \lambda_r \xi_r)$, depending on whether the problem is one of maximization or minimization, respectively. This involves a single variable search for λ_r along each of the n directions, $p_{r-1} + \lambda_r \xi_r$.

The search procedure requires the interval of uncertainty, or the limits within which the value of λ_r is expected to lie, to be determined first. The Fibonacci method [4,7] is based on the repeated evaluation of two experiments placed symmetrically in the interval of uncertainty. On the assumption that the function is unimodal and from the values of the function obtained from these two experiments, a portion of the interval is eliminated and the procedure repeated over the remaining interval. The interval of uncertainty is thus progressively reduced until it is within tolerable limits.

The problem is one of locating the first experiment, since the second is placed symmetrically with respect to the first. This is done by using the Fibonacci series of numbers where the current values of the series is the sum of the preceding two values:

$$F_0 = 1, \quad F_1 = 1$$

$$F_K = F_{K-1} + F_{K-2}, \quad K = 2, 3, \ldots$$

Wilde [7] has shown that for a search involving n experiments or observations, the final interval of uncertainty, L_n, is inversely proportional to the n^{th} Fibonacci number when the original interval is of unit length. If ε is the least separation between two points x_1 and x_2 for which a

difference between $f(x_1)$ and $f(x_2)$ can be detected, then

$$L_n = \frac{1 + F_{n-2}\,\varepsilon}{F_n}$$

After ε and n are decided upon, the first experiment is located using the formula

$$x_1 = \frac{F_{n-1} + (-1)^n \varepsilon}{F_n} (b - a)$$

where (b - a) represents the length of the interval of uncertainty. The second experiment is then located symmetrically using the relation

$$x_2 = a + b - x_1 .$$

A portion of the interval can now be eliminated and the procedure repeated over the remaining interval. Fig. 13 shows the progress of a search in five experiments for a maximization problem, with $\varepsilon = 0.01$.

The computer flow-chart for the search procedure is given in Appendix B. The program is coded as a Fortran subroutine and included with the computer program for Powell's method in Appendix C.

The Fibonacci search procedure is a very efficient one-dimensional search method for finding the unconstrained maximum or minimum, requires no derivatives and makes no assumption about the function other than unimodality in the interval of uncertainty. As such, it can be usefully applied in Powell's method, which is devised for optimizing functions whose derivatives are difficult to calculate.

APPENDIX B

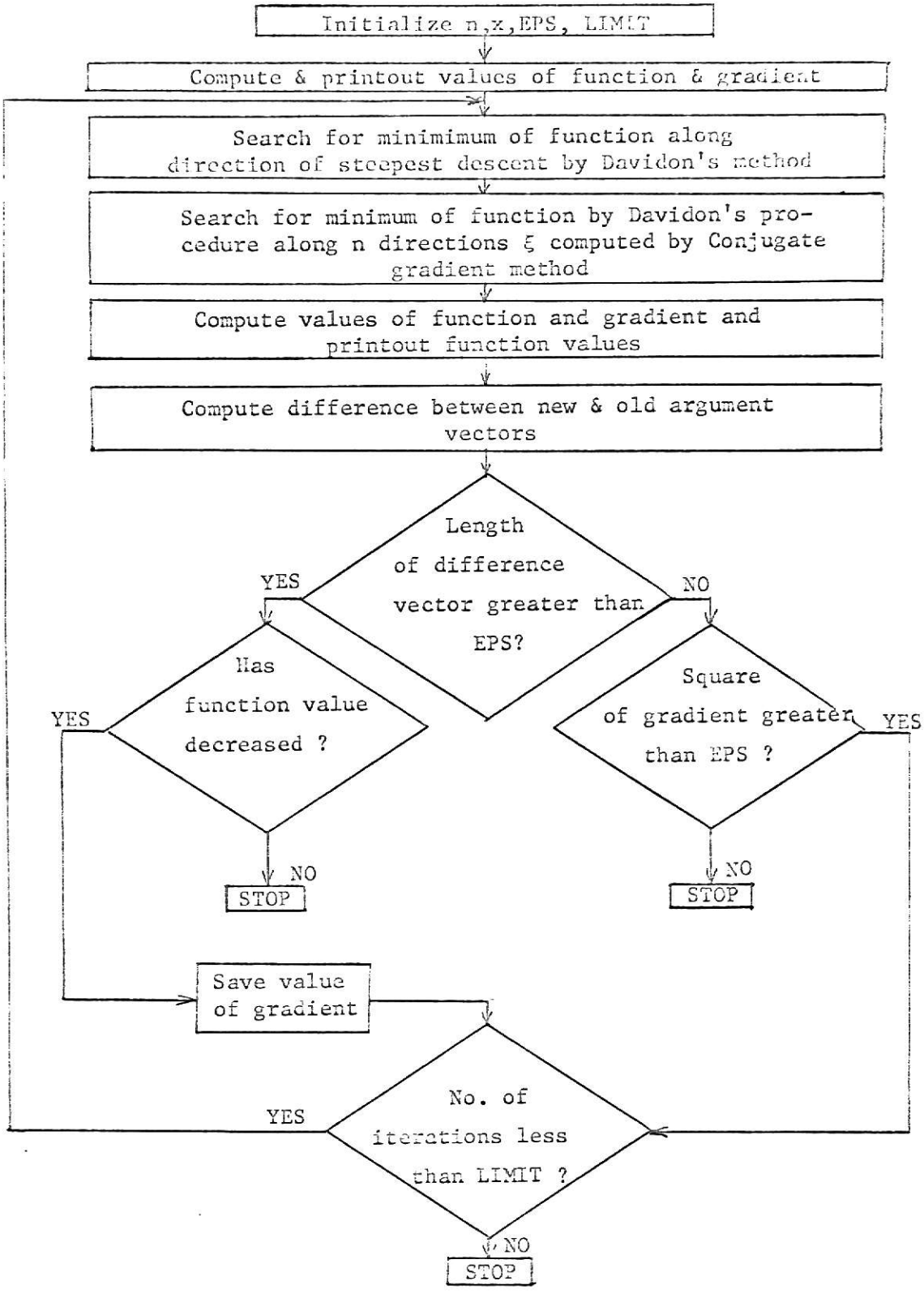


Fig. 9 Flowchart for minimization by the Fletcher & Reeves' method

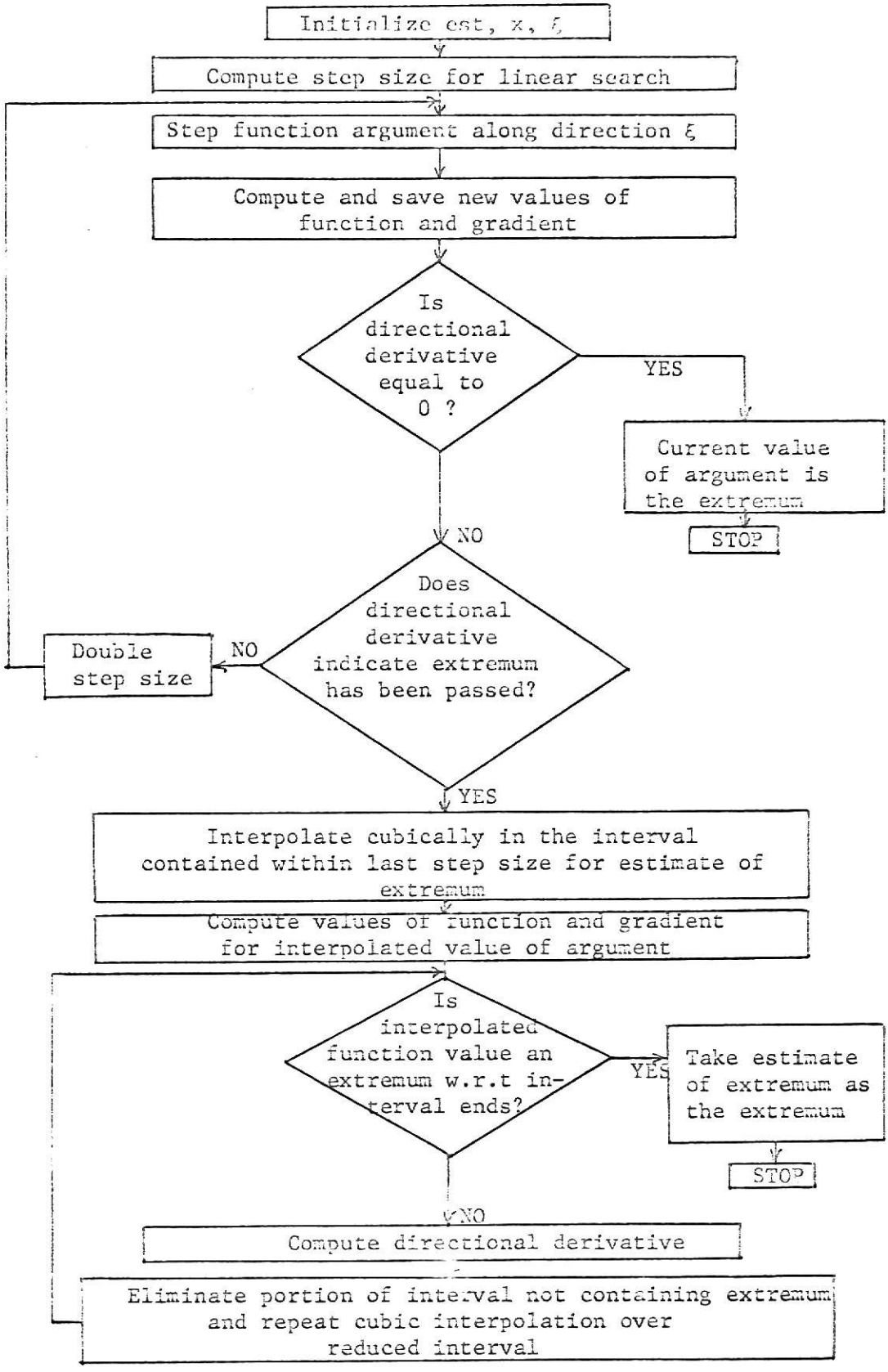


Fig. 10 Flowchart for Davidon's Linear Search procedure

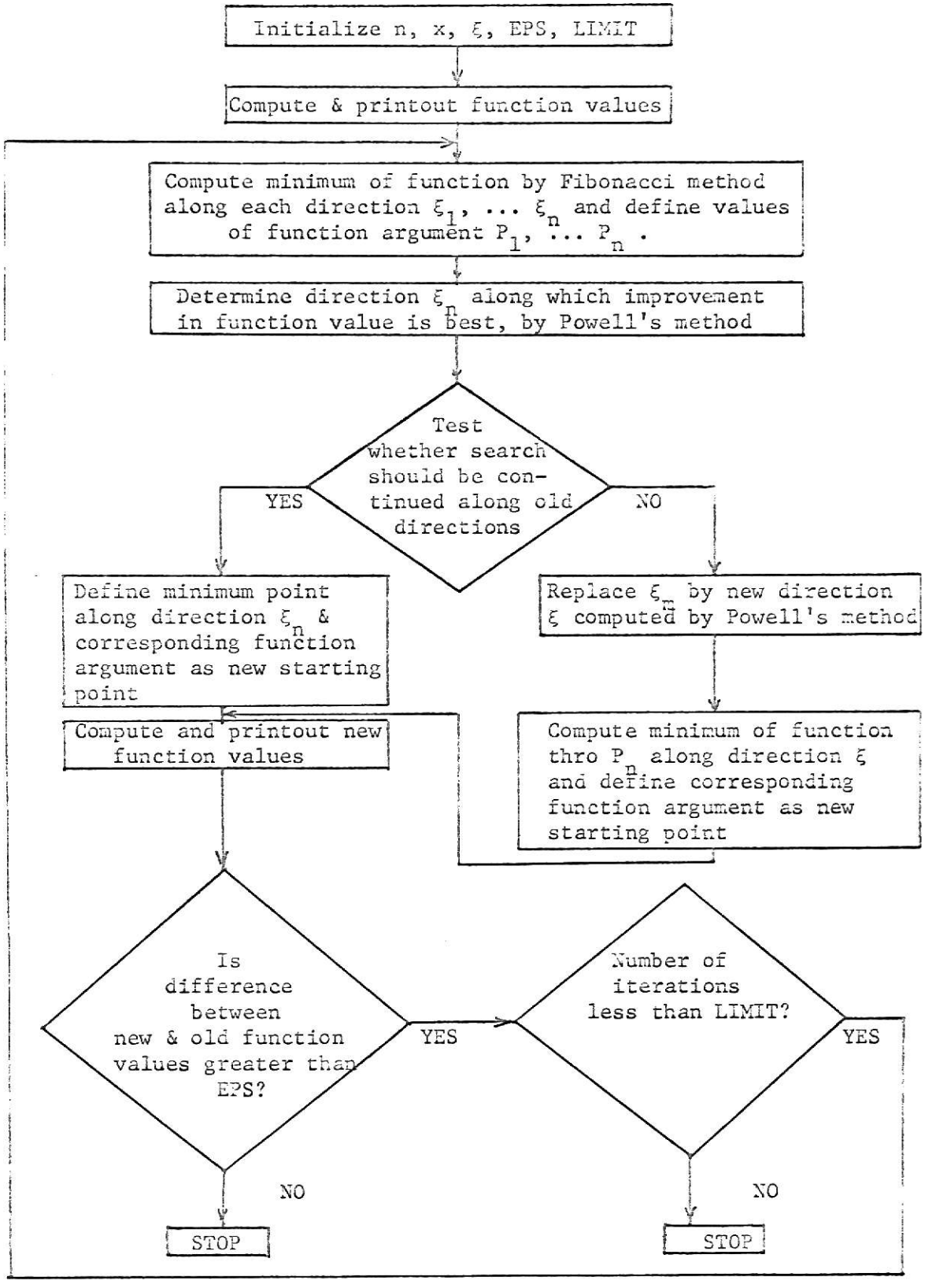


Fig. 11 Flow chart for minimization by the Powell's method

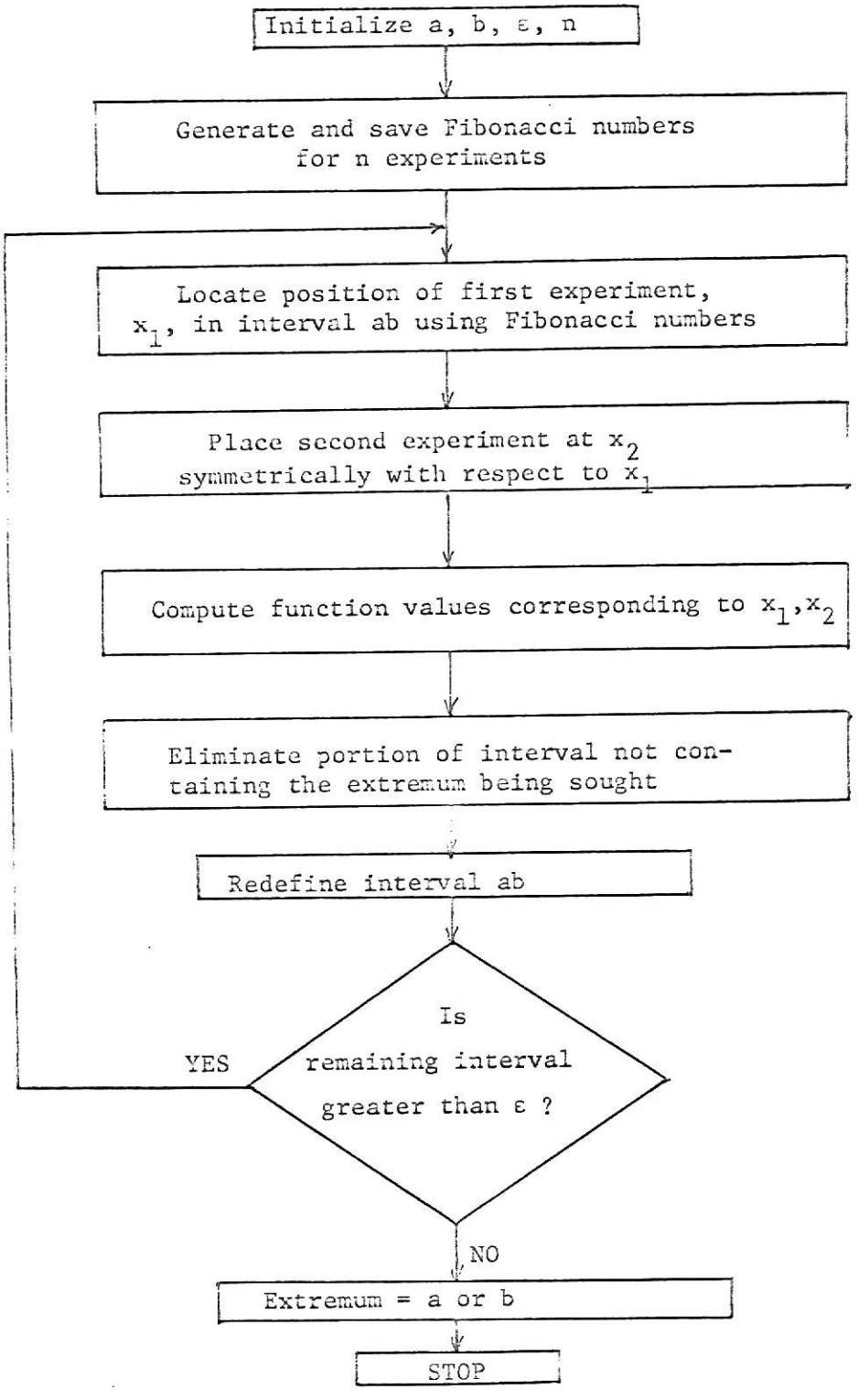


Fig. 12 Flowchart for the Fibonacci search procedure

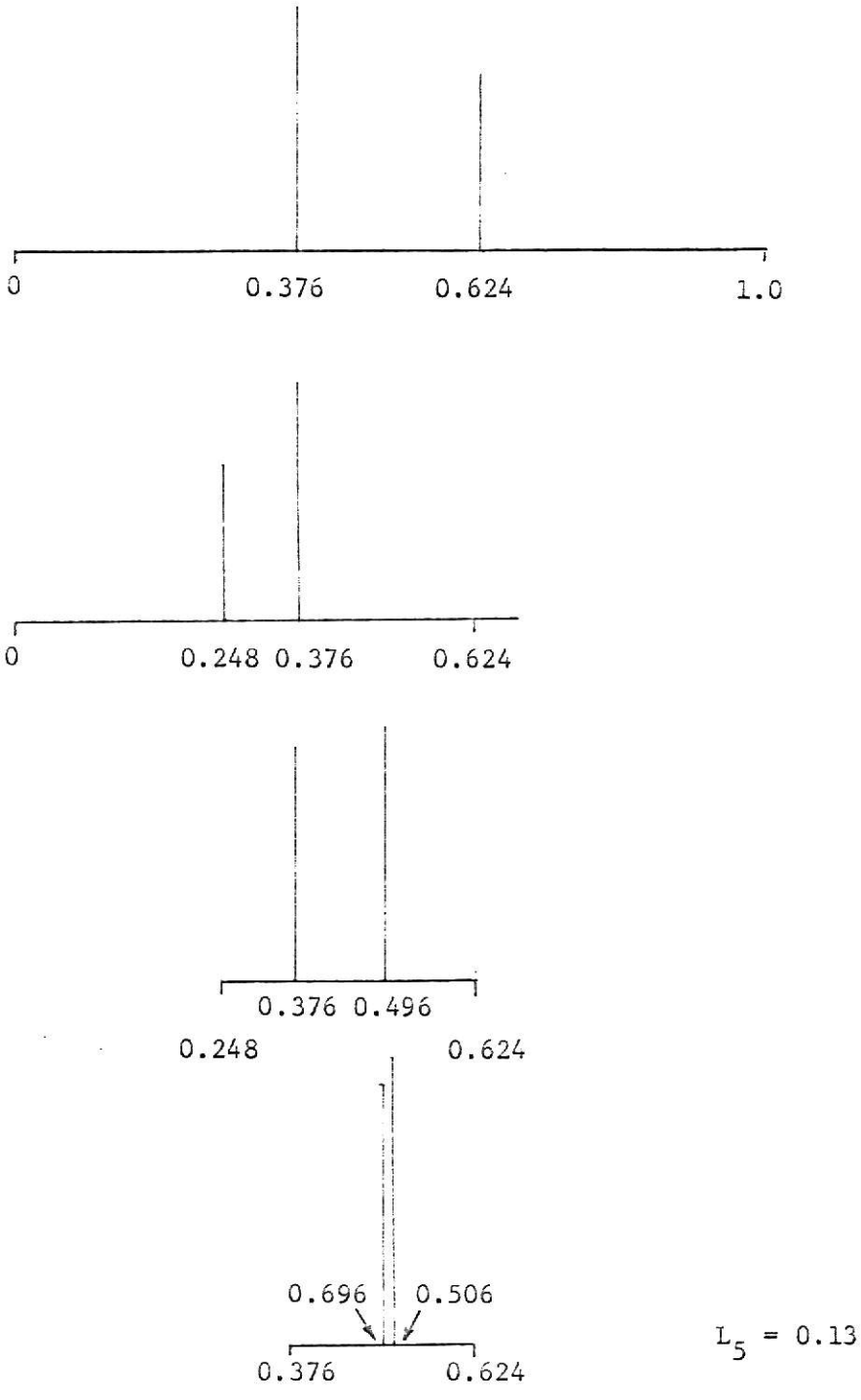


Fig. 13. Fibonacci search for 5 experiments with $\varepsilon = 0.01$ and unit interval of uncertainty

APPENDIX C

```
C       COMPUTER PROGRAM AND RESULTS FOR INVENTORY CONTROL PROBLEM WITH
C       ONE STATE VARIABLE BY THE CONJUGATE GRADIENT METHOD OF FLETCHER
C       AND REEVES
        EXTERNAL FUNCT
        DIMENSION X(20),G(20),H(20),XX(20),QS(20),S(20)
        N=10
        N1=N+1
        X(1)=1
        X(2)=1
        X(3)=1
        X(4)=1
        X(5)=1
        X(6)=1
        X(7)=1
        X(8)=1
        X(9)=1
        X(10)=1
        EST=1
        LIMIT=600
        EPS=10.**(-2)
        CALL FMCG(N,N1,X,F,G,EST,EPS,LIMIT,IER,H,XX,QS,S)
        STOP
        END
```

```
      SUBRCUTINE FUNCT(N,N1,X,F,G,XX,QS,S)
      DIMENSION X(20),G(20),XX(20),QS(20),S(20),RP(20),RX(20)
      DIMENSION RY(20),RW(20)
      AQ=2
      BQ=1
      C=5
      CI=0.1
      QS(1)=AQ
      XX(1)=C
      XM=10
      PM=5
      CP=0.001
      DT=0.1
      DO 1 I=2,N1
      QS(I)=AQ+BQ*(I-1)*DT
      XX(I)=XX(I-1)+(X(I-1)-QS(I))*DT
    1 CONTINUE
      DO 2 I=1,N
      RP(I)=(CP*(PM-X(I))*EXP((PM-X(I))**2))*(-2.)*DT
      RW(I)=XM-XX(I)/2.
    2 CONTINUE
      DO 4 I=2,N1
      RX(I)=2.*XM-3.*XX(I)/2.
      RY(I)=2.*XM-2.*XX(I)
    4 CONTINUE
      RZ=2.*XM-2.*XX(N)-XX(N1)
      DD=-CI*(DT**2)
      RL=2.*XM-3.*XX(N)/2.-XX(N1)
      RM=XM-XX(N)/2.-XX(N1)/2.
      S(1)=(CI*(XM-XX(2)/2.-XX(1)/2.)**2+CP*EXP((PM-X(1))**2))*DT
      DO 3 I=2,N
    3 S(I)=S(I-1)+(CI*(XM-XX(I+1)/2.-XX(I)/2.)**2+CP*EXP((PM-X(I))**2))*
     9DT
      F=S(N)
      G(1)=RP(1)+DD*(RW(1)+RX(2)+RY(3)+RY(4)+RY(5)+RY(6)+RY(7)+RY(8)+
     1RY(9)+RZ)
      G(2)=RP(2)+DD*(RW(2)+RX(3)+RY(4)+RY(5)+RY(6)+RY(7)+RY(8)+RY(9)+
     2RZ)
      G(3)=RP(3)+DD*(RW(3)+RX(4)+RY(5)+RY(6)+RY(7)+RY(8)+RY(9)+RZ)
      G(4)=RP(4)+DD*(RW(4)+RX(5)+RY(6)+RY(7)+RY(8)+RY(9)+RZ)
      G(5)=RP(5)+DD*(RW(5)+RX(6)+RY(7)+RY(8)+RY(9)+RZ)
      G(6)=RP(6)+DD*(RW(6)+RX(7)+RY(8)+RY(9)+RZ)
      G(7)=RP(7)+DD*(RW(7)+RX(8)+RY(9)+RZ)
      G(8)=RP(8)+DD*(RW(8)+RX(9)+RZ)
      G(9)=RP(9)+DD*(RW(9)+RL)
      G(10)=RP(10)+DD*(RW(10)+RM)
      RETURN
      END
```

```
      SUBROUTINE FMCG(N,N1,X,F,G,EST,EPS,LIMIT,IER,H,XX,QS,S)
      DIMENSION X(20),G(20),XX(20),QS(20),S(20),H(20)
      KOUNT=0
      CALL FUNCT(N,N1,X,F,G,XX,QS,S)
      PRINT 77
   77 FORMAT(' ',//////)
      PRINT 66
   66 FORMAT('                                        STAGE 1   STAGE 2   STAGE
     5 3   STAGE 4   STAGE 5   STAGE 6   STAGE 7   STAGE 8   STAGE 9   S
     6TAGE 10')
      PRINT 55,KOUNT,(X(I),I=1,N),(QS(I),I=2,N1),(XX(I),I=2,N1),(S(I),I=
     11,N)
   55 FORMAT('                  ITERATION NO.'I3,//'
     1  P = '10F10.2,/'                              Q = '10F10.2,/'
     2                X = '10F10.2,/'                          COST = '10F10
     2.2,/)
      IER=0
      N1=N+1
    1 DO 43 II=1,N1
      KOUNT=KOUNT+1
      OLDF=F
      GNRM=0.
      DO 2 J=1,N
    2 GNRM=GNRM+G(J)*G(J)
      IF(GNRM)46,46,3
    3 IF(II-1)4,4,6
    4 DO 5 J=1,N
    5 H(J)=-G(J)
      GO TO 8
    6 AMBDA=GNRM/OLDG
      DO 7 J=1,N
    7 H(J)=AMBDA*H(J)-G(J)
    8 DY=0.
      HNRM=0.
      DO 9 J=1,N
      K=J+N
      H(K)=X(J)
      HNRM=HNRM+ABS(H(J))
    9 DY=DY+H(J)*G(J)
      IF(DY)10,42,42
   10 SNRM=1./HNRM
      FY=F
      ALFA=2.*(EST-F)/DY
      AMBDA=SNRM
      IF(ALFA)13,13,11
   11 IF(ALFA-AMBDA)12,13,13
   12 AMBDA=ALFA
   13 ALFA=0.
   14 FX=FY
      DX=DY
```

```
      DO 15 I=1,N
   15 X(I)=X(I)+AMBDA*H(I)
      CALL FUNCT(N,N1,X,F,G,XX,QS,S)
      FY=F
      DY=0.
      DO 16 I=1,N
   16 DY=DY+G(I)*H(I)
      IF(DY)17,38,20
   17 IF(FY-FX)18,20,20
   18 AMBDA=AMBDA+ALFA
      ALFA=AMBDA
      IF(HNRM*AMBDA-1.E10)14,14,19
   19 IER=2
      PRINT 55,KOUNT,(X(I),I=1,N),(QS(I),I=2,N1),(XX(I),I=2,N1),(S(I),I=
     11,N)
      RETURN
   20 T=0.
   21 IF(AMBDA)22,38,22
   22 Z=3.*(FX-FY)/AMBDA+DX+DY
      ALFA=AMAX1(ABS(Z),ABS(DX),ABS(DY))
      DALFA=Z/ALFA
      DALFA=DALFA*DALFA-DX/ALFA*DY/ALFA
      IF(DALFA)23,27,27
   23 DO 24 J=1,N
      K=N+J
   24 X(J)=H(K)
      CALL FUNCT(N,N1,X,F,G,XX,QS,S)
   25 IF(IER)47,26,47
   26 IER=-1
      PRINT 55,KOUNT,(X(I),I=1,N),(QS(I),I=2,N1),(XX(I),I=2,N1),(S(I),I=
     11,N)
      GOTO 1
   27 W=ALFA*SQRT(DALFA)
      ALFA=(DY+W-Z)*AMBDA/(DY+2.*W-DX)
      DO 28 I=1,N
   28 X(I)=X(I)+(T-ALFA)*H(I)
      CALL FUNCT(N,N1,X,F,G,XX,QS,S)
      IF(F-FX)29,29,30
   29 IF(F-FY)38,38,30
   30 DALFA=0.
      DO 31 I=1,N
   31 DALFA=DALFA+G(I)*H(I)
      IF(DALFA)32,35,35
   32 IF(F-FX)34,33,35
   33 IF(DX-DALFA)34,38,34
   34 FX=F
      DX=DALFA
      T=ALFA
      AMBDA=ALFA
      GO TO 21
```

```
35 IF(FY-F)37,36,37
36 IF(DY-DALFA)37,38,37
37 FY=F
   DY=DALFA
   AMBDA=AMBDA-ALFA
   GO TO 20
38 T=0.
   DO 39 J=1,N
   K=J+N
   H(K)=X(J)-H(K)
39 T=T+ABS(H(K))
   IF(KOUNT-N1)41,40,40
40 IF(T-EPS)45,45,41
41 IF(OLDF-F+EPS)19,25,42
42 OLDG=GNRM
   PRINT 55,KOUNT,(X(I),I=1,N),(QS(I),I=2,N1),(XX(I),I=2,N1),(S(I),I=
  1,N)
   IF(KOUNT-LIMIT)43,44,44
43 IER=0
   GO TO 1
44 IER=1
   IF(GNRM-EPS)46,46,47
45 IF(GNRM-EPS)46,46,48
48 IF(IER.EQ.0) GO TO 49
   GO TO 47
46 IER=0
47 PRINT 55,KOUNT,(X(I),I=1,N),(QS(I),I=2,N1),(XX(I),I=2,N1),(S(I),I=
  1,N)
   RETURN
49 IER=-1
   PRINT 55,KOUNT,(X(I),I=1,N),(QS(I),I=2,N1),(XX(I),I=2,N1),(S(I),I=
  1,N)
   GO TO 1
   END
```

	STAGE 1	STAGE 2	STAGE 3	STAGE 4	STAGE 5	STAGE 6	STAGE 7	STAGE 8	STAGE 9	STAGE 10
ITERATION NO. 0										
P =	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
Q =	2.10	2.20	2.30	2.40	2.50	2.60	2.70	2.80	2.90	3.00
X =	4.89	4.77	4.64	4.50	4.35	4.19	4.02	3.84	3.65	3.45
COST =	888.87	1777.74	2666.64	3555.54	4444.46	5333.40	6222.36	7111.33	8000.33	8889.36
ITERATION NO. 1										
P =	6.44	6.44	6.44	6.44	6.44	6.44	6.44	6.44	6.44	6.44
Q =	2.10	2.20	2.30	2.40	2.50	2.60	2.70	2.80	2.90	3.00
X =	5.43	5.86	6.27	6.68	7.07	7.45	7.83	8.19	8.55	8.89
COST =	0.23	0.42	0.58	0.70	0.80	0.88	0.93	0.97	1.00	1.02
ITERATION NO. 2										
P =	7.25	7.11	6.98	6.87	6.76	6.67	6.59	6.53	6.47	6.51
Q =	2.10	2.20	2.30	2.40	2.50	2.60	2.70	2.80	2.90	3.00
X =	5.52	6.01	6.47	6.92	7.35	7.75	8.14	8.52	8.87	9.22
COST =	0.24	0.43	0.58	0.69	0.77	0.83	0.88	0.91	0.92	0.93
ITERATION NO. 3										
P =	7.15	7.19	7.11	6.99	6.87	6.75	6.64	6.55	6.48	6.54
Q =	2.10	2.20	2.30	2.40	2.50	2.60	2.70	2.80	2.90	3.00
X =	5.51	6.00	6.49	6.94	7.38	7.80	8.19	8.57	8.92	9.28
COST =	0.24	0.43	0.58	0.69	0.77	0.83	0.88	0.90	0.92	0.93
ITERATION NO. 4										
P =	7.16	7.12	7.12	7.06	6.95	6.82	6.69	6.58	6.48	6.57
Q =	2.10	2.20	2.30	2.40	2.50	2.60	2.70	2.80	2.90	3.00
X =	5.51	6.00	6.48	6.95	7.39	7.81	8.21	8.59	8.95	9.31
COST =	0.24	0.43	0.58	0.69	0.78	0.84	0.88	0.90	0.92	0.93
ITERATION NO. 5										
P =	7.18	7.11	7.09	7.06	6.98	6.87	6.73	6.60	6.48	6.60
Q =	2.10	2.20	2.30	2.40	2.50	2.60	2.70	2.80	2.90	3.00
X =	5.51	6.00	6.48	6.94	7.39	7.82	8.22	8.60	8.96	9.32
COST =	0.24	0.43	0.58	0.69	0.78	0.84	0.88	0.90	0.92	0.93

ITERATION NO. 6

P =	7.18	7.13	7.07	7.03	6.99	6.89	6.76	6.62	6.49	6.62
Q =	2.10	2.20	2.30	2.40	2.50	2.60	2.70	2.80	2.90	3.00
X =	5.51	6.00	6.48	6.94	7.39	7.82	8.23	8.61	8.97	9.33
COST =	0.24	0.43	0.58	0.69	0.78	0.84	0.88	0.90	0.92	0.93

ITERATION NO. 7

P =	7.18	7.13	7.07	7.03	6.99	6.89	6.76	6.62	6.49	6.63
Q =	2.10	2.20	2.30	2.40	2.50	2.60	2.70	2.80	2.90	3.00
X =	5.51	6.00	6.48	6.94	7.39	7.82	8.23	8.61	8.97	9.33
COST =	0.24	0.43	0.58	0.69	0.78	0.84	0.88	0.90	0.92	0.93

ITERATION NO. 8

P =	7.18	7.13	7.07	7.03	6.98	6.90	6.77	6.62	6.49	6.63
Q =	2.10	2.20	2.30	2.40	2.50	2.60	2.70	2.80	2.90	3.00
X =	5.51	6.00	6.48	6.94	7.39	7.82	8.23	8.61	8.97	9.33
COST =	0.24	0.43	0.58	0.69	0.78	0.84	0.88	0.90	0.92	0.93

ITERATION NO. 9

P =	7.18	7.13	7.07	7.03	6.98	6.90	6.77	6.62	6.49	6.63
Q =	2.10	2.20	2.30	2.40	2.50	2.60	2.70	2.80	2.90	3.00
X =	5.51	6.00	6.48	6.94	7.39	7.82	8.23	8.61	8.97	9.33
COST =	0.24	0.43	0.58	0.69	0.78	0.84	0.88	0.90	0.92	0.93

ITERATION NO. 10

P =	7.18	7.13	7.07	7.03	6.98	6.90	6.77	6.62	6.49	6.63
Q =	2.10	2.20	2.30	2.40	2.50	2.60	2.70	2.80	2.90	3.00
X =	5.51	6.00	6.48	6.94	7.39	7.82	8.23	8.61	8.97	9.33
COST =	0.24	0.43	0.58	0.69	0.78	0.84	0.88	0.90	0.92	0.93

ITERATION NO. 11

P =	7.17	7.13	7.08	7.03	6.98	6.90	6.77	6.62	6.49	6.63
Q =	2.10	2.20	2.30	2.40	2.50	2.60	2.70	2.80	2.90	3.00
X =	5.51	6.00	6.48	6.94	7.39	7.82	8.23	8.61	8.97	9.33
COST =	0.24	0.43	0.58	0.69	0.77	0.84	0.88	0.90	0.92	0.93

ITERATION NO. 12

P =	7.17	7.13	7.08	7.03	6.98	6.90	6.77	6.62	6.49	6.63
Q =	2.10	2.20	2.30	2.40	2.50	2.60	2.70	2.80	2.90	3.00
X =	5.51	6.00	6.48	6.94	7.39	7.82	8.23	8.61	8.97	9.33
COST =	0.24	0.43	0.58	0.69	0.77	0.84	0.88	0.90	0.92	0.93

```
C       COMPUTER PROGRAM AND RESULTS FOR INVENTORY CONTROL PROBLEM WITH
C       ONE STATE VARIABLE BY THE METHOD OF POWELL
        EXTERNAL FUNCT,FIBON
        DIMENSION A(20),QS(20),X(20),S(20)
        N=5
        A(1)=1
        A(2)=1
        A(3)=1
        A(4)=1
        A(5)=1
        LIMIT=600
        EPS=10.**(-2)
        CALL POWELL(N,A,EPS,LIMIT,KOUNT,N1,QS,X,S)
        STOP
        END
```

```
      SUBROUTINE FUNCT(N,N1,A,QS,X,S)
      DIMENSION A(20),QS(20),X(20),S(20)
      AQ=2
      BQ=1
      C=5
      CI=0.1
      QS(1)=AQ
      X(1)=C
      XM=10
      PM=5
      CP=0.001
      DT=0.2
      DO 1 I=2,N1
      QS(I)=AQ+BQ*(I-1)*DT
      X(I)=X(I-1)+(A(I-1)-QS(I))*DT
    1 CONTINUE
      S(1)=(CI*(XM-X(2)/2.-X(1)/2.)**2+CP*EXP((PM-A(1))**2))*DT
      DO 2 I=2,N
      S(I)=S(I-1)+(CI*(XM-X(I+1)/2.-X(I)/2.)**2+CP*EXP((PM-A(I))**2))*DT
    2 CONTINUE
      RETURN
      END
```

```
      SUBROUTINE FIBON(Q,N,N1,J,E,ELL)
      DIMENSION FIB(50),Q(20),X1(20),X2(20),E(20,20),QS(20)
      DIMENSION X(20),S(20)
      ERR=10.**(-2)
      FIB(1)=1.
      FIB(2)=2.
      DO 1 L=3,20
      FIB(L)=FIB(L-1)+FIB(L-2)
    1 CONTINUE
      A=-5
      B=5
      DO 8 L=1,20
      K=21-L
      W=-1**K
      EL1=((FIB(K-1)+W*ERR)/FIB(K))*(B-A)+A
      EL2=A+B-EL1
      DO 2 I=1,N
      X1(I)=Q(I)+EL1*E(I,J)
      X2(I)=Q(I)+EL2*E(I,J)
    2 CONTINUE
      CALL FUNCT(N,N1,X1,QS,X,S)
      S1=S(N)
      CALL FUNCT(N,N1,X2,QS,X,S)
      S2=S(N)
      IF(S1-S2) 5,3,3
    3 B=EL1
      GO TO 7
    5 A=EL2
    7 IF(ABS(B-A).LE.ERR) GO TO 9
    8 CONTINUE
    9 ELL=A
      RETURN
      END
```

```
      SUBROUTINE POWELL(N,A,EPS,LIMIT,KOUNT,N1,QS,X,S)
      DIMENSION A(20),E(20,20),Q(20),EL(20),P(20,20),Y(20),FA(20)
      DIMENSION X(20),QS(20),S(20),DEL(20),R(20)
      N1=N+1
      DO 1 J=1,N
      DO 1 I=1,N
      E(I,J)=0
    1 CONTINUE
      DO 2 I=1,N
      E(I,I)=1
    2 CONTINUE
      KOUNT=0
      CALL FUNCT(N,N1,A,QS,X,S)
      PRINT 77
   77 FORMAT(' ',//////)
      PRINT 44
   44 FORMAT('                                        STAGE 1    STAGE 2    STAGE
     5 3    STAGE 4    STAGE 5')
      PRINT 22,KOUNT,(A(I),I=1,N),(QS(I),I=2,N1),(X(I),I=2,N1),(S(I),I=
     21,N)
   22 FORMAT('                ITERATION NO.'I3,//'
     1  P = '5F10.2,/'                        Q = '5F10.2,/'
     2             X = '5F10.2,/'                      COST = '5F10.2,
     4/)
   33 CONTINUE
      DO 3 I=1,N
      P(I,1)=A(I)
    3 CONTINUE
      CALL FUNCT(N,N1,A,QS,X,S)
      FA(1)=S(N)
      DO 6 K=1,N
      DO 4 I=1,N
      J=K
    4 Q(I)=P(I,J)
      CALL FIBON(Q,N,N1,J,E,ELL)
      EL(J)=ELL
      DO 5 I=1,N
      P(I,J+1)=P(I,J)+EL(J)*E(I,J)
    5 R(I)=P(I,J+1)
      CALL FUNCT(N,N1,R,QS,X,S)
      FA(K+1)=S(N)
    6 CONTINUE
      DO 7 K=1,N
    7 DEL(K)=FA(K)-FA(K+1)
      DMAX=AMAX1(DEL(1),DEL(2),DEL(3),DEL(4),DEL(5))
      DO 8 K=1,N
      IF(DMAX.EQ.DEL(K)) GO TO 9
    8 CONTINUE
    9 J=K
      DO 13 I=1,N
```

```
      Y(I)=2.*P(I,N1)-P(I,1)
   13 CONTINUE
      CALL FUNCT(N,N1,Y,QS,X,S)
      FY=S(N)
      IF(FY.GE.FA(1)) GO TO 17
      QQ=(FA(1)-2.*FA(N1)+FY)*((FA(1)-FA(N1)-DMAX)**2)
      RR=((FA(1)-FY)**2)*(DMAX/2.)
      IF(QQ.GE.RR) GO TO 17
      DO 14 I=1,N
   14 E(I,J)=P(I,N1)-P(I,1)
      DO 15 I=1,N
   15 Q(I)=P(I,N1)
      CALL FIBON(Q,N,N1,J,E,ELL)
      EL(J)=ELL
      DO 16 I=1,N
   16 A(I)=P(I,N1)+EL(J)*E(I,J)
      CALL FUNCT(N,N1,A,QS,X,S)
      FNEW=S(N)
      GO TO 19
   17 CONTINUE
      DO 18 I=1,N
   18 A(I)=P(I,N1)
      CALL FUNCT(N,N1,A,QS,X,S)
      FNEW=S(N)
   19 KOUNT=KOUNT+1
      PRINT 22,KOUNT,(A(I),I=1,N),(QS(I),I=2,N1),(X(I),I=2,N1),(S(I),I=
     21,N)
      IF(ABS(FA(1)-FNEW)-EPS) 21,21,20
   20 IF(KOUNT-LIMIT)33,33,21
   21 RETURN
      END
```

	STAGE 1	STAGE 2	STAGE 3	STAGE 4	STAGE 5
ITERATION NO. 0					
P =	1.00	1.00	1.00	1.00	1.00
Q =	2.20	2.40	2.60	2.80	3.00
X =	4.76	4.48	4.16	3.80	3.40
COST =	1777.75	3555.55	5333.41	7111.36	8889.39
ITERATION NO. 1					
P =	5.96	5.95	5.98	5.95	5.99
Q =	2.20	2.40	2.60	2.80	3.00
X =	5.75	6.46	7.14	7.77	8.36
COST =	0.43	0.73	0.94	1.07	1.14
ITERATION NO. 2					
P =	7.18	7.08	6.95	6.76	6.41
Q =	2.20	2.40	2.60	2.80	3.00
X =	6.00	6.93	7.80	8.60	9.28
COST =	0.43	0.69	0.84	0.91	0.93
ITERATION NO. 3					
P =	7.15	7.06	6.94	6.76	6.42
Q =	2.20	2.40	2.60	2.80	3.00
X =	5.99	6.92	7.79	8.58	9.27
COST =	0.43	0.69	0.84	0.91	0.93

```
C       COMPUTER PROGRAM AND RESULTS FOR INVENTORY AND ADVERTISING PROBLEM
C       WITH TWO STATE VARIABLES BY THE METHOD OF FLETCHER AND REEVES
        EXTERNAL FUNCT
        DIMENSION X(20),G(20),H(20),XX(20),QS(20),PP(20),S(20)
        N=5
        N1=N+1
        X(1)=8
        X(2)=5
        X(3)=1
        X(4)=1
        X(5)=1
        EST=-600
        LIMIT=600
        EPS=10.**(-2)
        CALL FMCG(N,N1,X,F,G,EST,EPS,LIMIT,IER,H,XX,QS,PP,S)
        STOP
        END
```

```
      SUBROUTINE FUNCT(N,N1,X,F,G,XX,QS,PP,S)
      DIMENSION X(20),G(20),XX(20),QS(20),PP(20),S(20)
      DIMENSION RC(20),RP(20),RX(20),RT(20),RQ(20),RS(20),RF(20)
      XX(1)=20
      QS(1)=20
      F=10
      AP=70
      BP=100
      PP(1)=AP
      PN=150
      CA=1.5
      CI=0.15
      PI=50
      C=2
      DT=0.2
      DO 1 I=2,N1
      PP(I)=AP+BP*(I-1)*DT
      QS(I)=(QS(I-1)*(1+(C+X(I-1))*DT))/(1+(C+X(I-1))*QS(I-1)*DT/PN)
      XX(I)=XX(I-1)+(PP(I-1)-QS(I-1))*DT
    1 CONTINUE
      S(1)=(F*QS(2)-CI*(PI-XX(2))**2-CA*X(1)*QS(2))*DT
      DO 2 I=2,N
      S(I)=S(I-1)+(F*QS(I+1)-CI*(PI-XX(I+1))**2-CA*X(I)*QS(I+1))*DT
    2 CONTINUE
      F=-S(N)
      DO 3 I=2,N1
      RC(I)=-CA*QS(I)*DT
      RP(I)=PI-XX(I)
    3 CONTINUE
      DC=-2.*CI*DT
      DO 4 I=1,N
      RT(I)=(1+(C+X(I))*DT)/((1+(C+X(I))*QS(I)*DT/PN)**2)
      RQ(I)=(QS(I)*(DT**2)*(1-QS(I)/PN))/((1+(C+X(I))*QS(I)*DT/PN)**2)
      RF(I)=F-CA*X(I)
    4 CONTINUE
      G(1)=RC(2)+RQ(1)*(RF(1)+RF(2)*RT(2)+RF(3)*RT(3)*RT(2)+RF(4)*RT(4)*
     1RT(3)*RT(2)+RF(5)*RT(5)*RT(4)*RT(3)*RT(2)+DC*(RP(3)+RP(4)*(1+RT(2)
     2)+RP(5)*(1+RT(2)+RT(2)*RT(3))+RP(6)*(1+RT(2)+RT(2)*RT(3)+RT(2)*
     3RT(3)*RT(4))))
      G(2)=RC(3)+RQ(2)*(RF(2)+RF(3)*RT(3)+RF(4)*RT(4)*RT(3)+RF(5)*RT(5)*
     4RT(4)*RT(3)+DC*(RP(4)+RP(5)*(1+RT(3))+RP(6)*(1+RT(3)+RT(3)*RT(4)
     5)))
      G(3)=RC(4)+RQ(3)*(RF(3)+RF(4)*RT(4)+RF(5)*RT(5)*RT(4)+DC*(RP(5)+
     4RP(6)*(1+RT(4))))
      G(4)=RC(5)+RQ(4)*(RF(4)+RF(5)*RT(5)+DC*RP(6))
      G(5)=RC(6)+RQ(5)*RF(5)
      DO 5 I=1,N
      G(I)=-G(I)
    5 CONTINUE
      RETURN
      END
```

```
      SUBROUTINE FMCG(N,N1,X,F,G,EST,EPS,LIMIT,IER,H,XX,QS,PP,S)
      DIMENSION X(20),G(20),H(20),S(20),XX(20),QS(20),PP(20)
      KOUNT=0
      CALL FUNCT(N,N1,X,F,G,XX,QS,PP,S)
      PRINT 77
   77 FORMAT(' ',//////)
      PRINT 66
   66 FORMAT('                                          STAGE 1    STAGE 2    STAGE
     5 3    STAGE 4    STAGE 5')
      PRINT 55,KOUNT,(X(I),I=1,N),(PP(I),I=2,N1),(QS(I),I=2,N1),(XX(I),I
     2=2,N1),(S(I),I=1,N)
   55 FORMAT('                  ITERATION NO.'I3,//'
     2  A = '5F10.2,/'                                P = '5F10.2,/'
     3               Q = '5F10.2,/'                                X = '5F10.2,/
     4'                       PROFIT = '5F10.2,/)
      IER=0
      N1=N+1
    1 DO 43 II=1,N1
      KOUNT=KOUNT+1
      OLDF=F
      GNRM=0.
      DO 2 J=1,N
    2 GNRM=GNRM+G(J)*G(J)
      IF(GNRM)46,46,3
    3 IF(II-1)4,4,6
    4 DO 5 J=1,N
    5 H(J)=-G(J)
      GO TO 8
    6 AMBDA=GNRM/OLDG
      DO 7 J=1,N
    7 H(J)=AMBDA*H(J)-G(J)
    8 DY=0.
      HNRM=0.
      DO 9 J=1,N
      K=J+N
      H(K)=X(J)
      HNRM=HNRM+ABS(H(J))
    9 DY=DY+H(J)*G(J)
      IF(DY)10,42,42
   10 SNRM=1./HNRM
      FY=F
      ALFA=2.*(EST-F)/DY
      AMBDA=SNRM
      IF(ALFA)13,13,11
   11 IF(ALFA-AMBDA)12,13,13
   12 AMBDA=ALFA
   13 ALFA=0.
   14 FX=FY
      DX=DY
      DO 15 I=1,N
```

```
15 X(I)=X(I)+AMBDA*H(I)
   CALL FUNCT(N,N1,X,F,G,XX,QS,PP,S)
   FY=F
   DY=0.
   DO 16 I=1,N
16 DY=DY+G(I)*H(I)
   IF(DY)17,38,20
17 IF(FY-FX)18,20,20
18 AMBDA=AMBDA+ALFA
   ALFA=AMBDA
   IF(HNRM*AMBDA-1.E10)14,14,19
19 IER=2
   PRINT 55,KOUNT,(X(I),I=1,N),(PP(I),I=2,N1),(QS(I),I=2,N1),(XX(I),I
  2=2,N1),(S(I),I=1,N)
   RETURN
20 T=0.
21 IF(AMBDA)22,38,22
22 Z=3.*(FX-FY)/AMBDA+DX+DY
   ALFA=AMAX1(ABS(Z),ABS(DX),ABS(DY))
   DALFA=Z/ALFA
   DALFA=DALFA*DALFA-DX/ALFA*DY/ALFA
   IF(DALFA)23,27,27
23 DO 24 J=1,N
   K=N+J
24 X(J)=H(K)
   CALL FUNCT(N,N1,X,F,G,XX,QS,PP,S)
25 IF(IER)47,26,47
26 IER=-1
   PRINT 55,KOUNT,(X(I),I=1,N),(PP(I),I=2,N1),(QS(I),I=2,N1),(XX(I),I
  2=2,N1),(S(I),I=1,N)
   GOTO 1
27 W=ALFA*SQRT(DALFA)
   ALFA=(DY+W-Z)*AMBDA/(DY+2.*W-DX)
   DO 28 I=1,N
28 X(I)=X(I)+(T-ALFA)*H(I)
   CALL FUNCT(N,N1,X,F,G,XX,QS,PP,S)
   IF(F-FX)29,29,30
29 IF(F-FY)38,38,30
30 DALFA=0.
   DO 31 I=1,N
31 DALFA=DALFA+G(I)*H(I)
   IF(DALFA)32,35,35
32 IF(F-FX)34,33,35
33 IF(DX-DALFA)34,38,34
34 FX=F
   DX=DALFA
   T=ALFA
   AMBDA=ALFA
   GO TO 21
35 IF(FY-F)37,36,37
```

```
36 IF(DY-DALFA)37,38,37
37 FY=F
   DY=DALFA
   AMBDA=AMBDA-ALFA
   GO TO 20
38 T=0.
   DO 39 J=1,N
   K=J+N
   H(K)=X(J)-H(K)
39 T=T+ABS(H(K))
   IF(KOUNT-N1)41,40,40
40 IF(T-EPS)45,45,41
41 IF(OLDF-F+EPS)19,25,42
42 OLDG=GNRM
   PRINT 55,KOUNT,(X(I),I=1,N),(PP(I),I=2,N1),(QS(I),I=2,N1),(XX(I),I
  2=2,N1),(S(I),I=1,N)
   IF(KOUNT-LIMIT)43,44,44
43 IER=0
   GO TO 1
44 IER=1
   IF(GNRM-EPS)46,46,47
45 IF(GNRM-EPS)46,46,48
48 IF(IER.EQ.0) GO TO 49
   GO TO 47
46 IER=0
47 PRINT 55,KOUNT,(X(I),I=1,N),(PP(I),I=2,N1),(QS(I),I=2,N1),(XX(I),I
  2=2,N1),(S(I),I=1,N)
   RETURN
49 IER=-1
   PRINT 55,KOUNT,(X(I),I=1,N),(PP(I),I=2,N1),(QS(I),I=2,N1),(XX(I),I
  2=2,N1),(S(I),I=1,N)
   GO TO 1
   END
```

	STAGE 1	STAGE 2	STAGE 3	STAGE 4	STAGE 5
ITERATION NO. 0					
A =	8.00	5.00	1.00	1.00	1.00
P =	90.00	110.00	130.00	150.00	170.00
Q =	47.37	78.83	95.89	110.89	122.91
X =	30.00	38.53	44.76	51.58	59.40
PROFIT =	-30.95	4.52	166.71	355.16	561.45
ITERATION NO. 1					
A =	6.78	3.79	-0.26	0.23	0.64
P =	90.00	110.00	130.00	150.00	170.00
Q =	44.67	71.68	82.84	96.11	109.74
X =	30.00	39.07	46.73	56.16	66.94
PROFIT =	-13.57	44.69	216.57	401.01	590.73
ITERATION NO. 2					
A =	6.78	3.78	-0.27	0.23	0.64
P =	90.00	110.00	130.00	150.00	170.00
Q =	44.65	71.64	82.74	96.00	109.64
X =	30.00	39.07	46.74	56.19	66.99
PROFIT =	-13.47	44.92	216.81	401.18	590.74
ITERATION NO. 3					
A =	6.78	3.78	-0.27	0.22	0.64
P =	90.00	110.00	130.00	150.00	170.00
Q =	44.65	71.63	82.74	95.99	109.63
X =	30.00	39.07	46.74	56.20	67.00
PROFIT =	-13.46	44.94	216.84	401.19	590.74
ITERATION NO. 4					
A =	6.77	3.78	-0.27	0.22	0.64
P =	90.00	110.00	130.00	150.00	170.00
Q =	44.65	71.63	82.73	95.98	109.62
X =	30.00	39.07	46.74	56.20	67.00
PROFIT =	-13.45	44.96	216.86	401.21	590.74
ITERATION NO. 5					

A =	6.77	3.78	-0.28	0.22	0.64
P =	90.00	110.00	130.00	150.00	170.00
Q =	44.64	71.61	82.69	95.93	109.58
X =	30.00	39.07	46.75	56.21	67.02
PROFIT =	-13.41	45.05	216.96	401.28	590.74

ITERATION NO. 6

A =	6.77	3.78	-0.28	0.22	0.64
P =	90.00	110.00	130.00	150.00	170.00
Q =	44.64	71.61	82.69	95.93	109.58
X =	30.00	39.07	46.75	56.21	67.02
PROFIT =	-13.41	45.05	216.96	401.28	590.74

ITERATION NO. 7

A =	6.77	3.78	-0.28	0.22	0.64
P =	90.00	110.00	130.00	150.00	170.00
Q =	44.64	71.59	82.65	95.89	109.53
X =	30.00	39.07	46.75	56.22	67.05
PROFIT =	-13.37	45.14	217.06	401.35	590.74

ITERATION NO. 8

A =	6.77	3.78	-0.28	0.22	0.64
P =	90.00	110.00	130.00	150.00	170.00
Q =	44.64	71.59	82.65	95.89	109.53
X =	30.00	39.07	46.75	56.22	67.05
PROFIT =	-13.37	45.14	217.06	401.35	590.74

ITERATION NO. 9

A =	6.77	3.77	-0.28	0.22	0.64
P =	90.00	110.00	130.00	150.00	170.00
Q =	44.64	71.59	82.64	95.88	109.53
X =	30.00	39.07	46.76	56.23	67.05
PROFIT =	-13.36	45.16	217.08	401.36	590.74

```
C       FUNCTION MAXIMATION BY METHOD OF POWELL
C       INVENTORY CONTROL PROBLEM WITH TWO STATE VARIABLES
        EXTERNAL FUNCT,FIBON
        DIMENSION A(20),QS(20),X(20),PP(20),S(20)
        N=10
        A(1)=5
        A(2)=5
        A(3)=5
        A(4)=5
        A(5)=5
        A(6)=5
        A(7)=5
        A(8)=5
        A(9)=5
        A(10)=5
        LIMIT=600
        EPS=10.**(-2)
        CALL POWELL(N,A,EPS,LIMIT,KOUNT,N1,QS,X,PP,S)
        STOP
        END
```

```
      SUBROUTINE FUNCT(N,N1,A,QS,X,PP,S)
      DIMENSION A(20),QS(20),X(20),PP(20),S(20)
      X(1)=20
      QS(1)=20
      F=10
      AP=70
      BP=100
      PP(1)=AP
      PN=150
      CA=1.5
      CI=0.15
      PI=50
      C=2
      DT=0.1
      DO 1 I=2,N1
      PP(I)=AP+BP*(I-1)*DT
      QS(I)=(QS(I-1)*(1+(C+A(I-1))*DT))/(1+(C+A(I-1))*QS(I-1)*DT/PN)
      X(I)=X(I-1)+(PP(I)-QS(I))*DT
    1 CONTINUE
      S(1)=(F*QS(2)-CI*(PI-X(2))**2-CA*A(1)*QS(2))*DT
      DO 2 I=2,N
      S(I)=S(I-1)+(F*QS(I+1)-CI*(PI-X(I+1))**2-CA*A(I)*QS(I+1))*DT
    2 CONTINUE
      RETURN
      END
```

```
      SUBROUTINE FIBON(Q,N,N1,J,E,ELL)
      DIMENSION FIB(50),Q(20),X1(20),X2(20),E(20,20),QS(20)
      DIMENSION X(20),PP(20),S(20)
      ERR=10.**(-2)
      FIB(1)=1.
      FIB(2)=2.
      DO 1 L=3,20
      FIB(L)=FIB(L-1)+FIB(L-2)
    1 CONTINUE
      A=-1
      B=50
      DO 8 L=1,20
      K=21-L
      W=-1**K
      EL1=((FIB(K-1)+W*ERR)/FIB(K))*(B-A)+A
      EL2=A+B-EL1
      DO 2 I=1,N
      X1(I)=Q(I)+EL1*E(I,J)
      X2(I)=Q(I)+EL2*E(I,J)
    2 CONTINUE
      CALL FUNCT(N,N1,X1,QS,X,PP,S)
      S1=S(N)
      CALL FUNCT(N,N1,X2,QS,X,PP,S)
      S2=S(N)
      IF(S1-S2) 3,5,5
    3 B=EL1
      GO TO 7
    5 A=EL2
    7 IF(ABS(B-A).LE.ERR) GO TO 9
    8 CONTINUE
    9 ELL=A
      RETURN
      END
```

```
      SUBROUTINE POWELL(N,A,EPS,LIMIT,KOUNT,N1,QS,X,PP,S)
      DIMENSION A(20),E(20,20),Q(20),EL(20),P(20,20),Y(20),FA(20)
      DIMENSION X(20),QS(20),PP(20),S(20),DEL(20),R(20)
      N1=N+1
      DO 1 J=1,N
      DO 1 I=1,N
      E(I,J)=0
    1 CONTINUE
      DO 2 I=1,N
      E(I,I)=1
    2 CONTINUE
      KOUNT=0
      CALL FUNCT(N,N1,A,QS,X,PP,S)
      PRINT 77
   77 FORMAT(' ',//////)
      PRINT 44
   44 FORMAT('                                        STAGE 1   STAGE 2   STAGE
     5 3   STAGE 4   STAGE 5   STAGE 6   STAGE 7   STAGE 8   STAGE 9   S
     6TAGE 10')
      PRINT 22,KOUNT,(A(I),I=1,N),(PP(I),I=2,N1),(QS(I),I=2,N1),(X(I),I=
     12,N1),(S(I),I=1,N)
   22 FORMAT('                ITERATION NO.'I3,//'
     1  A = '10F10.2,/'                        P = '10F10.2,/'
     2                Q = '10F10.2,/'                          X = '10F10
     3.2,/'                    PROFIT = '10F10.2,/)
   33 CONTINUE
      DO 3 I=1,N
      P(I,1)=A(I)
    3 CONTINUE
      CALL FUNCT(N,N1,A,QS,X,PP,S)
      FA(1)=S(N)
      DO 6 K=1,N
      DO 4 I=1,N
      J=K
    4 Q(I)=P(I,J)
      CALL FIBON(Q,N,N1,J,E,ELL)
      EL(J)=ELL
      DO 5 I=1,N
      P(I,J+1)=P(I,J)+EL(J)*E(I,J)
    5 R(I)=P(I,J+1)
      CALL FUNCT(N,N1,R,QS,X,PP,S)
      FA(K+1)=S(N)
    6 CONTINUE
      DO 7 K=1,N
    7 DEL(K)=FA(K+1)-FA(K)
      DMAX=AMAX1(DEL(1),DEL(2),DEL(3),DEL(4),DEL(5),DEL(6),DEL(7),DEL(8)
     1,DEL(9),DEL(10))
      DO 8 K=1,N
      IF(DMAX.EQ.DEL(K)) GO TO 9
    8 CONTINUE
```

```
 9 J=K
   DO 13 I=1,N
   Y(I)=2.*P(I,N1)-P(I,1)
13 CONTINUE
   CALL FUNCT(N,N1,Y,QS,X,PP,S)
   FY=S(N)
   IF(FY.LE.FA(1)) GO TO 17
   QQ=(FA(1)-2.*FA(N1)+FY)*((FA(1)-FA(N1)-DMAX)**2)
   RR=((FA(1)-FY)**2)*(DMAX/2.)
   IF(QQ.LE.RR) GO TO 17
   DO 14 I=1,N
14 E(I,J)=P(I,N1)-P(I,1)
   DO 15 I=1,N
15 Q(I)=P(I,N1)
   CALL FIBON(Q,N,N1,J,E,ELL)
   EL(J)=ELL
   DO 16 I=1,N
16 A(I)=P(I,N1)+EL(J)*E(I,J)
   CALL FUNCT(N,N1,A,QS,X,PP,S)
   FNEW=S(N)
   GO TO 19
17 CONTINUE
   DO 18 I=1,N
18 A(I)=P(I,N1)
   CALL FUNCT(N,N1,A,QS,X,PP,S)
   FNEW=S(N)
19 KOUNT=KOUNT+1
   PRINT 22,KOUNT,(A(I),I=1,N),(PP(I),I=2,N1),(QS(I),I=2,N1),(X(I),I=
  12,N1),(S(I),I=1,N)
   IF(ABS(FA(1)-FNEW)-EPS) 21,21,20
20 IF(KOUNT-LIMIT)33,33,21
21 RETURN
   END
```

	STAGE 1	STAGE 2	STAGE 3	STAGE 4	STAGE 5	STAGE 6	STAGE 7	STAGE 8	STAGE 9	STAGE 10
ITERATION NO. 0										
A =	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00	5.00
P =	80.00	90.00	100.00	110.00	120.00	130.00	140.00	150.00	160.00	170.00
Q =	31.10	46.17	64.57	84.35	102.90	118.18	129.49	137.21	142.21	145.31
X =	24.89	29.27	32.82	35.38	37.09	38.27	39.33	40.60	42.38	44.85
PROFIT =	-1.68	3.41	15.13	33.01	56.24	83.72	114.38	147.36	182.04	217.97
ITERATION NO. 1										
A =	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00
P =	80.00	90.00	100.00	110.00	120.00	130.00	140.00	150.00	160.00	170.00
Q =	29.63	42.38	57.98	75.31	92.60	108.11	120.76	130.28	137.04	141.63
X =	25.04	29.80	34.00	37.47	40.21	42.40	44.32	46.29	48.59	51.43
PROFIT =	2.50	13.34	32.69	60.46	96.06	138.44	186.26	238.17	292.95	349.57
ITERATION NO. 2										
A =	4.04	3.00	3.00	3.00	3.00	3.00	3.00	3.00	3.00	3.00
P =	80.00	90.00	100.00	110.00	120.00	130.00	140.00	150.00	160.00	170.00
Q =	29.69	40.52	53.55	68.16	83.31	97.80	110.64	121.24	129.52	135.70
X =	25.03	29.98	34.62	38.81	42.48	45.70	48.63	51.51	54.56	57.99
PROFIT =	2.35	18.62	44.53	80.14	125.11	178.62	239.45	306.10	377.02	450.69
ITERATION NO. 3										
A =	7.47	3.34	2.00	2.00	2.00	2.00	2.00	2.00	2.00	2.00
P =	80.00	90.00	100.00	110.00	120.00	130.00	140.00	150.00	160.00	170.00
Q =	34.58	47.23	58.72	71.08	83.66	95.76	106.79	116.37	124.33	130.72
X =	24.54	28.82	32.95	36.84	40.47	43.90	47.22	50.58	54.15	58.08
PROFIT =	-13.91	2.94	39.69	86.84	144.04	210.51	285.15	366.60	453.38	543.90
ITERATION NO. 4										
A =	10.42	4.68	1.84	1.00	1.00	1.00	1.00	1.00	1.00	1.00
P =	80.00	90.00	100.00	110.00	120.00	130.00	140.00	150.00	160.00	170.00
Q =	38.47	54.78	66.50	76.30	86.05	95.44	104.19	112.09	119.03	124.99
X =	24.15	27.68	31.03	34.40	37.79	41.25	44.83	48.62	52.72	57.22
PROFIT =	-31.69	-22.81	19.90	81.10	152.01	231.99	320.15	415.40	516.46	621.92
ITERATION NO. 5										

A =	12.49	6.14	2.56	0.53	0.00	0.00	0.00	0.00	0.00	0.00
P =	80.00	90.00	100.00	110.00	120.00	130.00	140.00	150.00	160.00	170.00
Q =	41.05	60.89	74.80	83.21	89.88	96.32	102.43	108.14	113.42	118.22
X =	23.90	26.81	29.33	32.01	35.02	38.39	42.14	46.33	50.99	56.17
PROFIT =	-46.05	-49.28	-9.59	62.19	148.71	243.00	344.50	452.44	565.84	683.50

ITERATION NO. 6

A =	13.77	7.41	3.58	0.98	-0.60	-1.00	-1.00	-1.00	-1.00	-1.00
P =	80.00	90.00	100.00	110.00	120.00	130.00	140.00	150.00	160.00	170.00
Q =	42.58	65.22	81.78	91.30	95.90	99.15	102.30	105.35	108.28	111.09
X =	23.74	26.22	28.04	29.91	32.32	35.41	39.18	43.64	48.81	54.71
PROFIT =	-55.68	-71.39	-40.81	31.08	130.98	241.81	357.70	478.24	602.74	730.16

ITERATION NO. 7

A =	14.32	8.30	4.52	1.78	-0.26	-1.61	-2.00	-2.00	-2.00	-2.00
P =	80.00	90.00	100.00	110.00	120.00	130.00	140.00	150.00	160.00	170.00
Q =	43.23	67.68	86.39	97.77	103.08	104.31	104.31	104.31	104.31	104.31
X =	23.68	25.91	27.27	28.49	30.19	32.75	36.32	40.89	46.46	53.03
PROFIT =	-60.04	-85.34	-65.25	-0.57	100.67	225.69	358.49	492.85	628.26	763.73

ITERATION NO. 8

A =	14.25	8.76	5.23	2.53	0.40	-1.35	-2.58	-3.00	-3.00	-3.00
P =	80.00	90.00	100.00	110.00	120.00	130.00	140.00	150.00	160.00	170.00
Q =	43.15	68.41	88.65	101.61	108.37	110.23	108.47	105.24	101.86	98.35
X =	23.68	25.84	26.98	27.82	28.98	30.96	34.11	38.59	44.40	51.57
PROFIT =	-59.50	-89.73	-78.61	-22.96	72.34	199.46	346.06	496.70	643.92	786.50

ITERATION NO. 9

A =	13.85	8.86	5.58	3.11	1.01	-0.78	-2.37	-3.60	-4.00	-4.00
P =	80.00	90.00	100.00	110.00	120.00	130.00	140.00	150.00	160.00	170.00
Q =	42.69	68.02	88.99	103.19	111.23	114.46	113.41	108.36	101.33	93.73
X =	23.73	25.93	27.03	27.71	28.59	30.14	32.80	36.97	42.83	50.46
PROFIT =	-56.37	-87.44	-80.84	-33.31	54.14	176.00	325.36	489.75	651.11	801.07

ITERATION NO. 10

A =	13.37	8.68	5.67	3.39	1.43	-0.29	-1.90	-3.45	-4.79	-5.00
P =	80.00	90.00	100.00	110.00	120.00	130.00	140.00	150.00	160.00	170.00
Q =	42.11	67.00	88.18	103.06	112.01	116.30	116.57	112.31	102.36	90.09
X =	23.79	26.09	27.27	27.97	28.77	30.14	32.48	36.25	42.01	50.00
PROFIT =	-52.62	-81.47	-76.04	-32.69	48.54	164.05	309.18	476.83	651.78	809.45

ITERATION NO. 11

A =	12.96	8.52	5.65	3.44	1.61	-0.02	-1.56	-3.12	-4.74	-6.00

OPTIMIZATION OF MANAGEMENT SYSTEMS BY THE CONJUGATE GRADIENT METHODS

by

BALAKRISHNAN RADHAKRISHNAN NAIR

B.E., UNIVERSITY OF MADRAS

MADRAS, INDIA, 1959

AN ABSTRACT OF A MASTER'S REPORT

submitted in partial fulfillment of the

requirements for the degree

MASTER OF SCIENCE

Department of Industrial Engineering

KANSAS STATE UNIVERSITY

Manhattan, Kansas

1969

The difficulties in obtaining analytical solutions to non-linear optimization problems have led to the development of several direct methods of optimization in recent years. Among these, the methods utilizing the knowledge of the gradient of the function being optimized are perhaps the most important. These gradient methods have the advantage of being able to handle a large number of state variables and are easily programmed. Their chief drawback has been the slow convergence rate to the optimum.

Hestenes and Stiefel, in 1952, devised an ingenious scheme to accelerate the convergence of the gradient method. They used the accumulated knowledge of the behaviour of the function to generate new directions of search, called conjugate directions, at the end of each iteration. Using this scheme, Fletcher and Reeves developed the conjugate gradient method of optimizing an unconstrained function of several variables.

Powell developed a technique based on generating conjugate directions of search without evaluating derivatives. This method of optimization is particularly useful in cases where it is tedious or practically impossible to evaluate derivatives of the functions.

The purpose of this report is to apply the two techniques to industrial management systems and make a critical analysis of the results obtained and the computational procedure used.

First the two techniques are described in some detail. Two models in production and inventory control are then considered. The first model involves the minimization of production and inventory costs. The second model considers the effect of advertising on the sales rate in a production and inventory situation and involves the maximization of profit.

The purpose of this report is to apply the two techniques described above, to solve a few optimization problems in the field of production and inventory control. Though these methods have been developed about five years back, very little has been published regarding their application to industrial and management problems. The results obtained with both the methods are compared and a few conclusions drawn about their relative merits and demerits.