Office of Naval Research
Contract N00014-75-C-0648 NR-102
National Science Foundation Grant HES 73-10323

# MIN: AN INTERACTIVE EDUCATIONAL PROGRAM FOR FUNCTION MINIMIZATION


By

R. Muralidharan and R. K. Jain

December 1975

Technical Report No. 658



Division of Engineering and Applied Physics
Harvard University • Cambridge, Massachusetts

Unclassified

SECURITY CLASSIFICATION OF THIS PAGE *(When Data Entered)*

| REPORT DOCUMENTATION PAGE | | READ INSTRUCTIONS BEFORE COMPLETING FORM |
|---|---|---|
| 1. REPORT NUMBER<br>Technical Report 658 | 2. GOVT ACCESSION NO. | 3. RECIPIENT'S CATALOG NUMBER |
| 4. TITLE *(and Subtitle)*<br>MIN: AN INTERACTIVE EDUCATIONAL PROGRAM FOR FUNCTION MINIMIZATION | | 5. TYPE OF REPORT & PERIOD COVERED<br>Interim Report<br>6. PERFORMING ORG. REPORT NUMBER |
| 7. AUTHOR(s)<br>R. Muralidharan<br>R. K. Jain | | 8. CONTRACT OR GRANT NUMBER(s)<br>N00014-75-C-0648<br>NSF-HES73-10323 |
| 9. PERFORMING ORGANIZATION NAME AND ADDRESS<br>Division of Engineering and Applied Physics<br>Harvard University<br>Cambridge, Massachusetts | | 10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS |
| 11. CONTROLLING OFFICE NAME AND ADDRESS | | 12. REPORT DATE<br>December 1975<br>13. NUMBER OF PAGES<br>54 |
| 14. MONITORING AGENCY NAME & ADDRESS*(if different from Controlling Office)* | | 15. SECURITY CLASS. *(of this report)*<br>Unclassified<br>15a. DECLASSIFICATION/DOWNGRADING SCHEDULE |

16. DISTRIBUTION STATEMENT *(of this Report)*

This document has been approved for public release and sale; its distribution is unlimited. Reproduction in whole or in part is permitted by the U. S. Government.

17. DISTRIBUTION STATEMENT *(of the abstract entered in Block 20, if different from Report)*

18. SUPPLEMENTARY NOTES

19. KEY WORDS *(Continue on reverse side if necessary and identify by block number)*

Educational tool
Interactive program
Stopping criteria
Function minimization
Line search

20. ABSTRACT *(Continue on reverse side if necessary and identify by block number)*

MIN is an interactive computer program package for function minimization. It provides building blocks with which a user can construct a program to solve his problem. Moreover, it is an educational tool which helps users "learn" what is happening by interacting with the program while it is executing, by testing out his intuition, by switching algorithms between iterations, etc. Currently MIN provides a choice of four search direction algorithms (Davidson Fletcher Powell method, Parallel Tangent Method,

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE
S/N 0102-014-6601 |

Unclassified

SECURITY CLASSIFICATION OF THIS PAGE *(When Data Entered)*

20. Abstract continued

Fletcher Reeve's Conjugate Gradient Method, and Gradient Method) and five line search algorithms (Golden Section, False Position, High Order, Fibbonacci and DSC-Powell Search). Also, it provides several stopping criterion, different output formats, and proper input checking facilities. The package has been written in BASIC language implemented on WANG 2200 computers.

Office of Naval Research

Contract N00014-75-C-0648 NR-372-012

National Science Foundation Grant HES73-10323

# MIN: AN INTERACTIVE EDUCATIONAL PROGRAM FOR FUNCTION MINIMIZATION

By

R. Muralidharan and R. K. Jain

Technical Report No. 658



December 1975

The research reported in this document was made possible through support extended the Division of Engineering and Applied Physics, Harvard University by the U. S. Army Research Office, the U. S. Air Force Office of Scientific Research and the U. S. Office of Naval Research under the Joint Services Electronics Program by Contract N00014-75-C-0648 and by the National Science Foundation under Grant HES73-10323.

Division of Engineering and Applied Physics

Harvard University · Cambridge, Massachusetts

## ABSTRACT

MIN is an interactive computer program package for function minimization. It provides building blocks with which a user can construct a program to solve his problem. Moreover, it is an educational tool which helps users "learn" what is happening by interacting with the program while it is executing, by testing out his intuition, by switching algorithms between iterations, etc. Currently MIN provides a choice of four search direction algorithms (Davidon Fletcher Powell method, Parallel Tangent Method, Fletcher Reeve's Conjugate Gradient Method, and Gradient Method) and five line search algorithms (Golden Section, False Position, High Order, Fibbonacci and DSC-Powell Search). Also, it provides several stopping criterion, different output formats, and proper input checking facilities. The package has been written in BASIC language implemented on WANG 2200 computers.

# 1. INTRODUCTION

The purpose of this report is to describe MIN, an interactive computer program package written in WANG 2200 BASIC.† MIN provides the building blocks with which a user can construct a program to solve his unconstrained function minimization problem. In developing MIN, emphasis was placed not on just getting answers, but on solving the problems in a nice way. Thus, MIN is an educational tool which helps the user "learn" what is happening by interacting with the program while it is executing, by testing out his intuition, by switching algorithms between iterations and by trying out a gamut of other possibilities. Of course, MIN may be used as a straightforward "canned" function minimization program if desired.

The organization of this report is as follows. In Section 2 we shall describe how MIN is implemented and what facilities it provides. Section 3 gives briefly the theory of the various algorithms used in MIN. Those interested in simply using MIN rather than learning its details may do so by skipping directly to the Appendix (user's guide to MIN).

---

† BASIC on WANG 2200 differs from that on other computers mainly in File and other I/O handling commands.

## 2. IMPLEMENTATION OF MIN

### 2.1 Overview

The problem is to find the minimum of a function V0(x) where x is an n-vector and V0 has a gradient vector V1(x).† To solve this problem the user may write his own custom made program or he may use MIN to provide the building blocks for his program.

Any program written to solve the above problem will have a flow chart similar to that in Fig. 1. The central idea behind this flow chart is illustrated in Fig. 2. Here x* is the as yet unknown solution to the problem which the user wants the program to search for. The starting point for the search is the user's guestimate of x*. Point $x_k$ is the approximation to x* found after k iterations by the program. At $x_k$, the program uses some algorithm (e.g., Gradient, DFP, etc.) to provide the next search direction $s_k$. For example, in case of gradient algorithm, search direction $s_k$ is opposite that of the gradient vector $V1(x_k)$ at $x_k$. Along $s_k$ the cost function V0 is a function of a single parameter $\omega$, i.e., $V0 = V0(x_k + \omega s_k)$. At this point the program uses a line search algorithm (e.g., Golden Section, Fibonacci Search, Cubic Search, etc.) to find the best value of $\omega$ --that is, the value $\omega_k$ which gives minimum V0 along the search direction $s_k$. Another essential part of the customized program is the "Stopping Criterion" which causes the program to stop iterating. An example of commonly used stopping criterion is "Stop when gradient norm is reduced below a specified tolerance."

---

† We use V0 to denote the function and Vn to denote n-th derivative of V0. In particular, V1 denotes gradient vector.

## 2.2 Building Blocks Provided by MIN

MIN provides several building blocks with which the user can construct his own minimization program. Also, he may switch building blocks and compare the performance of different programs with respect to his problem. MIN provides a choice of three stopping criteria, four minimization (or search direction finding) algorithms, and five line search algorithms as follows:

The three stopping criteria are:

1. Stop when gradient norm is reduced below a specified tolerance
2. Stop when parameter improvement is below a specified tolerance
3. Stop after a specified number of iterations

The four minimization (search direction finding) algorithms are:

1. DFP (Davidon Fletcher Powell) method
2. Partan (Parallel tangent) method
3. Fletcher-Reeves conjugate gradient method
4. Gradient method

These algorithms are discussed in Section 3.2.

The five line search algorithms are

1. Golden section search
2. False position search
3. D.S.C. Powell search
4. High order search
5. Fibonacci search

These line search algorithms are discussed in Section 3.3.

The justification for each of the stopping criteria and circumstances under which one is preferred will be discussed in Section 3.1. The user

has a choice of specifying any of eight possible combinations of these three criteria. For example, he could instruct:

> "Stop when gradient norm is reduced below $10^{-5}$, however, I can't wait longer than 100 iterations and I don't care about parameter improvement."

Of course, MIN will not understand the above English statement, instead the user will have to provide MIN with appropriate numerical answers to some questions. How to properly instruct MIN is discussed in the Appendix.

## 2.3 Modules of MIN

Educational value of MIN is evident from the fact that the user may not only try any one of 160 ($= 8 \times 4 \times 5$) different custom made programs, but he may switch back and forth among them, compare, and learn. Furthermore, MIN has a modular structure so that it is very easy to extend its capability (e.g., adding a fifth minimization algorithm) without affecting its present form. MIN consists of four different modules each made up of one or more files as follows:

1. Input Module - one file named "INPUT"
2. Execution Module - 3 files named "DFP", "PARTAN", and "FR"*
3. Line Search Module - 5 files named "GS", "FP", "DSC", "HO", and "FS"
4. Output Module - one file named "OUTPUT".

---

* The Execution Module avoids a separate file for gradient method using the fact that DFP when restarted every iteration yields the gradient method.

MIN uses one file from each module to construct the user specified "customized minimization program". This is illustrated in Fig. 3, which also indicates the user interaction with MIN. The KEYIN facility will be discussed in the next section. The logical flow chart for MIN is shown in Fig. 4. The first LOAD and RUN causes the input module to execute. This module sets up a dialog with the user and asks him to specify his function, tolerances and his choice of various options. It has a built in error checking routine to check if the user's response is appropriate and if not, to ask him to respond correctly. The sample shown in Fig. 5 illustrates this point. Here a question mark (?) at the beginning of a line procedes user response.

After all the data is in, the user is given a chance to correct any of them. Once the user okays all the data, MIN replaces the input module by appropriate files from other modules to build the user specified program. This process commonly called "overlaying" is necessary due to memory limitations of the computer [1]. Figure 6 shows the memory map for MIN, that is, a conceptual "snapshot" of what the computer memory holds. Here the horizontal represents "time" and the vertical the "memory". Note the indicated times at which the snapshot changes. The second change (reloading of the input module) is achieved by the user through the KEYIN facility discussed next.

## 2.4 KEYIN Facility

This permits the user to interrupt the program at any point during execution and gives him an option to change any of the input data. This is achieved by pressing special function key "15"* on the WANG 2200

---

* An earlier version of MIN (version 3) allows the user to turn screen display of detailed line search results on and off by keying-in special function key "0". However, this has been eliminated from MIN version 6 due to memory limitations.

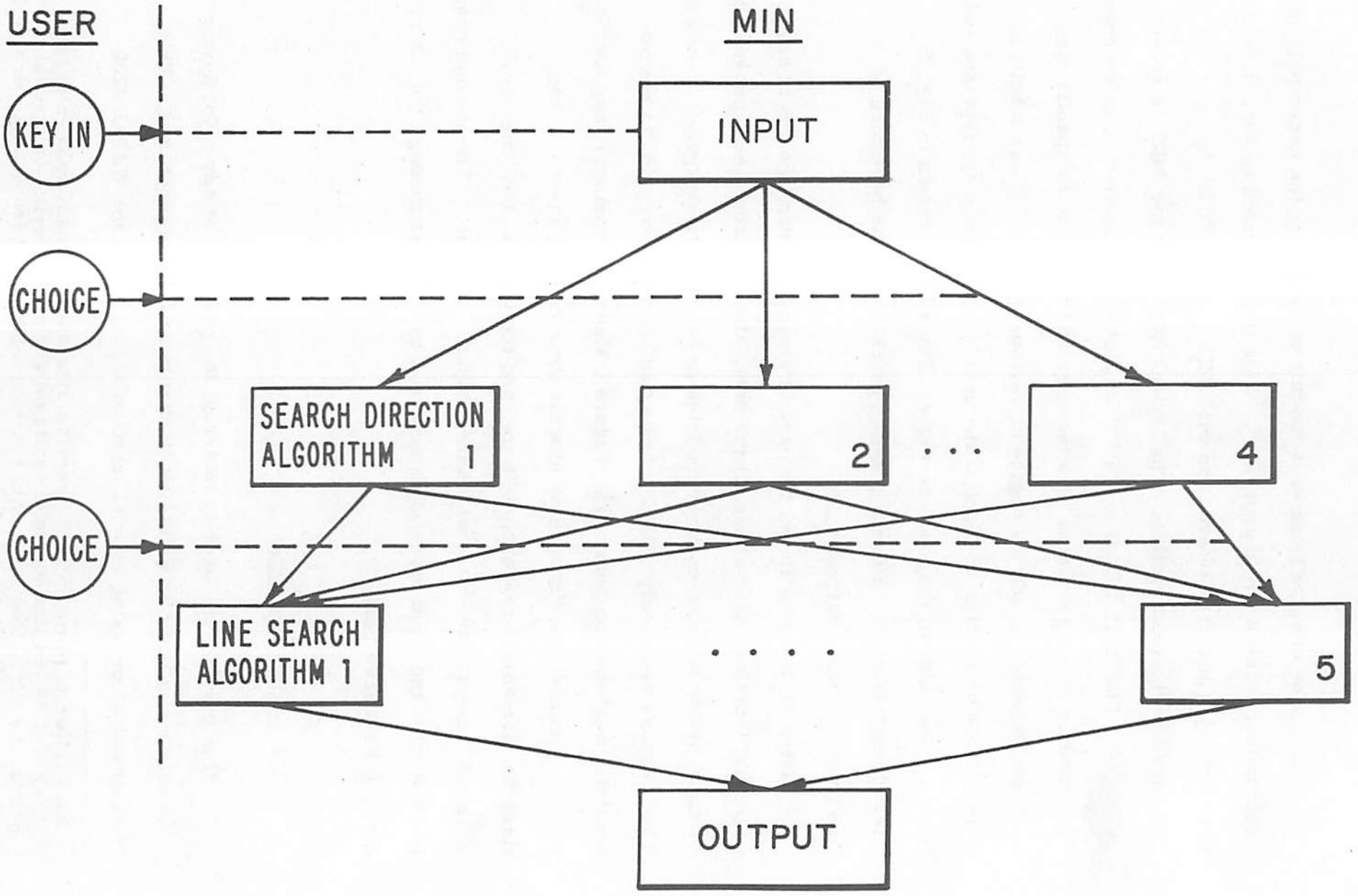


FIG. 3 BUILDING CUSTOMIZED MINIMIZATION PROGRAM WITH MIN

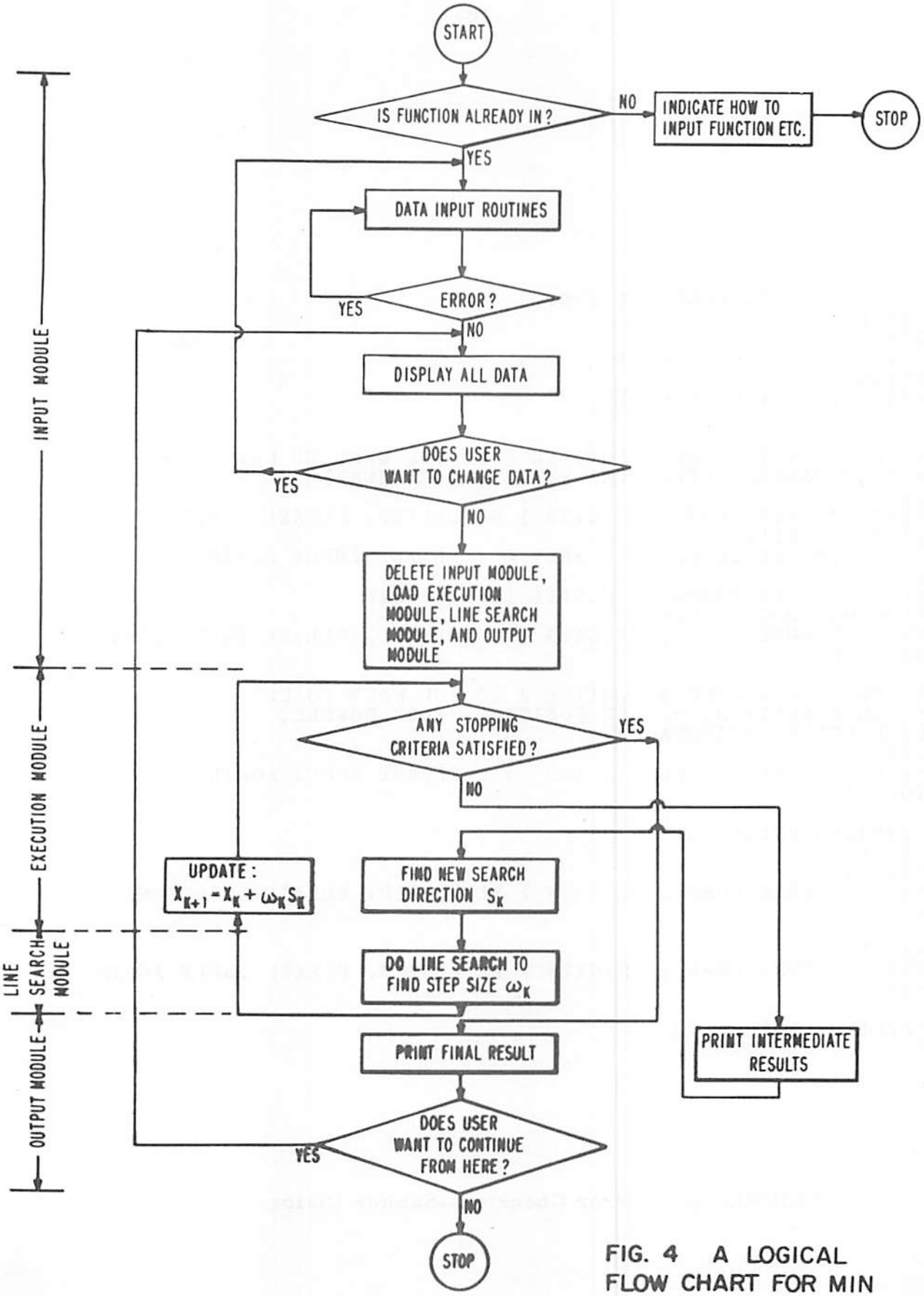


FIG. 4 A LOGICAL FLOW CHART FOR MIN

```
HAVE YOU STORED FUNCTION ETC.
(Y OR N)
? YES
PLEASE INPUT Y OR N ONLY
? CLEAR
PLEASE INPUT Y OR N ONLY
? Y

1.WHICH MINIMIZATION ALGORITHM # DO YOU WANT TO USE
(1=DFP,2=PARTAN,3=FLETCHER REEVES,4=GRADIENT)
? DFP
NUMERIC ANSWER (MAX 13 DIGITS ) REQUESTED. PLEASE INPUT AGAIN
? 1111111111111
ANSWER MUST BE BETWEEN  1 AND  4 . PLEASE INPUT AGAIN
? 1.5
ANSWER MUST BE INTEGER. PLEASE INPUT AGAIN
? 00000000000000001
NUMERIC ANSWER (MAX 13 DIGITS ) REQUESTED. PLEASE INPUT AGAIN
? 00000001

2.WHICH LINE SEARCH ALGORITHM # DO YOU WANT TO USE
(1=GOLDEN SECTION,2=FALSE POSITION,3=DSC POWELL,
4=HIGH ORDER,5=FIBONACCI)
? 1E5
ANSWER MUST BE BETWEEN  1 AND  5 . PLEASE INPUT AGAIN
? 100E-2

3.STARTING POINT
X( 1 )=
? UNKNOWN
NUMERIC ANSWER (MAX 13 DIGITS ) REQUESTED. PLEASE INPUT AGAIN
? 0
X( 2 )=
? -1END
NUMERIC ANSWER (MAX 13 DIGITS ) REQUESTED. PLEASE INPUT AGAIN
? -1E-5

4.GRADIENT TOLERANCE
? 0
```

FIGURE 5: Error Checking--Sample Dialog

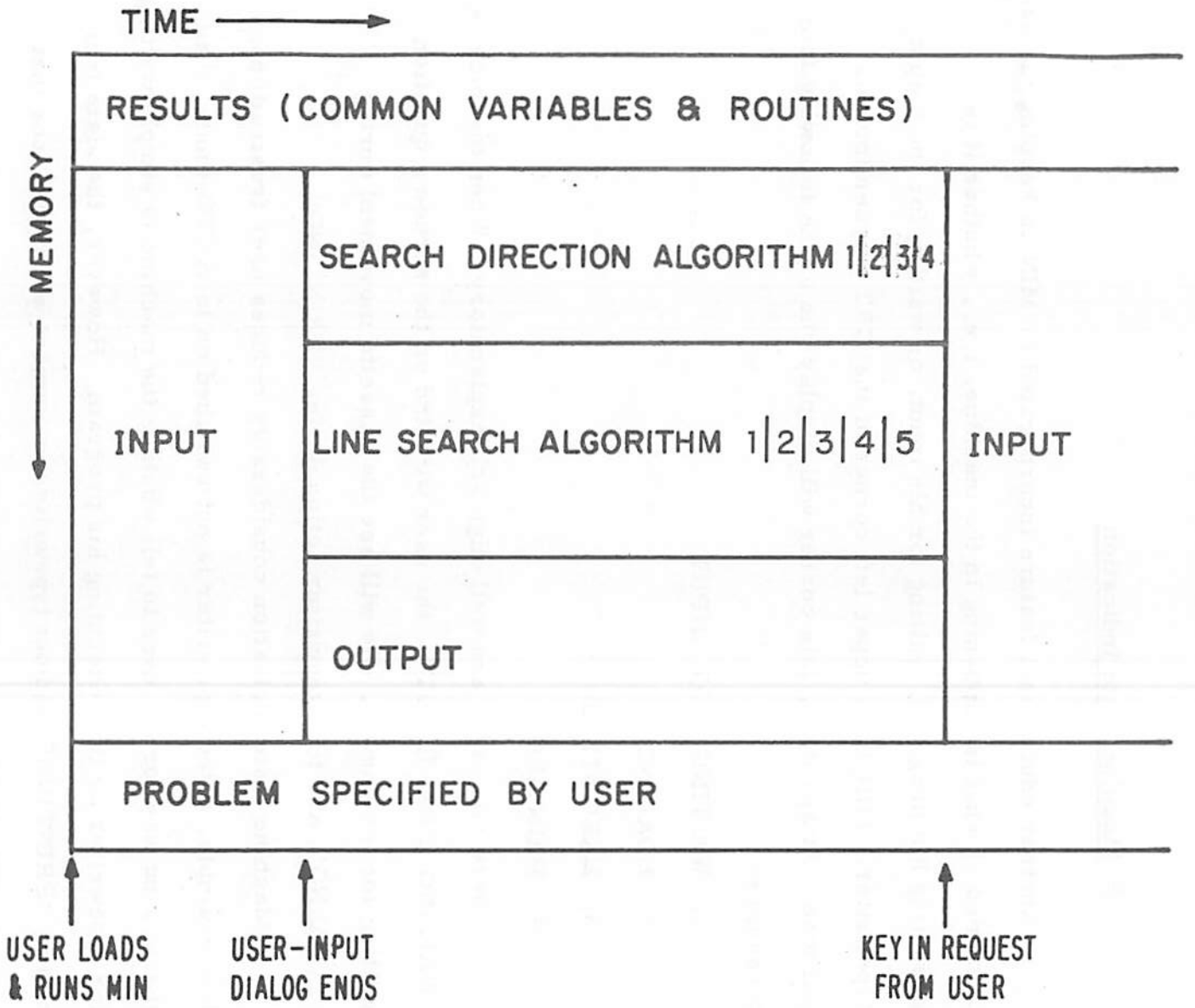


FIG. 6 MEMORY MAP FOR MIN

keyboard. When this happens MIN stops the program and asks him if he wants to change any data and if needed reloads the input module.

## 2.5 Machine State Indication

Another educational feature incorporated in MIN is keeping the user informed of what is happening in the machine, i.e., whether it is executing his program, waiting for his input, or waiting for the output typewriter. MIN uses upper left corner of the CRT screen for this purpose. At any time, this corner will display one of the following four messages:

1. WAITING FOR INPUT
2. LOADING
3. EXECUTING
4. PRINTING

These messages are well-nigh self explanatory. When machine is "WAITING FOR INPUT", the user will find on the screen a question waiting for his answer. He will see the cassette movement during "LOADING" and the typewriter action during "PRINTING".

Machine state indication considerably reduces user frustrations. For example, if the typewriter is not switched on to AUTO-mode, then there is no way for the user to tell whether the machine is simply waiting for typewriter or it is executing his program. However, the state indication "PRINTING" without typewriter movement warns the user that there is something wrong with the typewriter.

# 3. THEORETICAL DETAILS

## 3.1 Stopping Criteria

Choice of appropriate stopping criterion continues to be a topic of discussion among researchers in the field of optimization. One such lively discussion may be found in [2]. Recall from Section 2.2 that MIN provides a choice of any 8 possible combinations of 3 stopping criteria. If the function is very flat about the optimum (small hessian) then parameters may be quite inaccurate unless the gradient tolerance is very small (see Fig. 7). On the other hand, when minimizing a function with large hessian (e.g. penalty function), it is very difficult to reduce the gradient to small values and yet it is possible to obtain parameters quite accurately (see Fig. 8). This suggests that the bound (or tolerance) on gradient norm $\|V1\|$ must be, in some way, related to the hessian V2. After some experimentation it was found that for those interested in accurate parameter values, a bound on step size $\|x_{k+1}-x_k\|$ (which is proportional to $\|V2^{-1}\ V1\|$), would be an appropriate stopping criterion. This justifies the second stopping criterion. The third criterion of stopping after a certain number of iterations helps avoid frustrations of a user who has a limited time and is not very sure that his tolerances on gradient norm and step size will ever be met.

MIN accepts zero values for gradient and parameter tolerances. A zero value for parameter tolerance requires program to "stop when parameter improvement in any one iteration is below zero." It is obvious that step size (or similarly gradient norm) can never be negative, hence a zero tolerance is equivalent to nullifying corresponding stopping criterion.

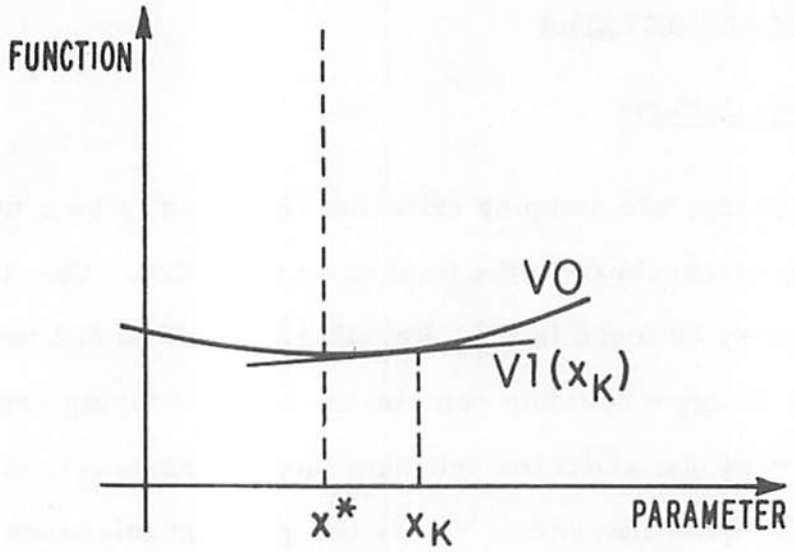


FIG. 7 SMALL GRADIENT TOLERANCE FOR FUNCTION WITH SMALL HESSIAN

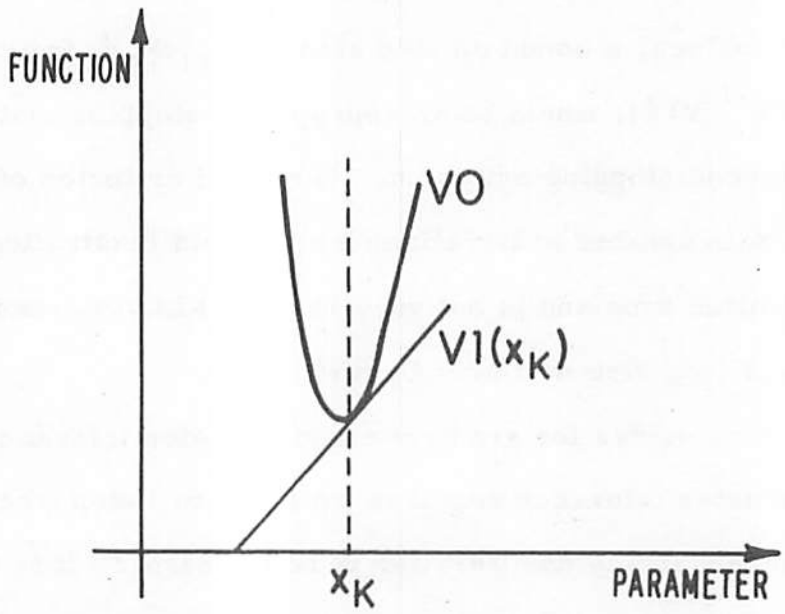


FIG. 8 LARGE GRADIENT TOLERANCE FOR FUNCTION WITH LARGE HESSIAN

To protect the user (from consequences of his bad choices) MIN uses two additional criteria to stop. These are 4 and 5 below:

4. Stop when MIN thinks it has reached the minimum. This happens when MIN finds that decrease even along steepest descent direction is less than $10^{-99}$, the least positive floating point number for WANG-2200.

5. Stop when MIN finds that

(a) the user specified search direction algorithm has failed to give a direction along which there is descent, and

(b) the user has instructed MIN not to restart search along steepest descent direction, under condition (a).

3.2 Search Direction Algorithms

3.2.1 DFP: This a quasi-Newton method originally proposed by Davidon [3] and subsequently developed by Fletcher and Powell [4]. It is also known as a "variable metric method". When V0 is quadratic it simultaneously generates the conjugate gradient directions while constructing the inverse hessian [5, p. 194]. The method for selecting $s_k$ is simply

$$s_k = - H_k g_k$$

where $g_k = V1(x_k)$, and $H_k$ is the approximation to the inverse Hessian updated by

$$H_{k+1} = H_k + \frac{p_k p_k^T}{p_k^T \Delta g_k} - \frac{H_k \Delta g_k \Delta g_k^T H_k}{\Delta g_k^T H_k \Delta g_k}$$

$$p_k \triangleq \omega_k s_k$$

$$\Delta g_k \triangleq g_{k+1} - g_k \quad .$$

When V0 is a quadratic with constant Hessian C it is easily shown that $H_n = C^{-1}$ and that the directions $s_k$ are C-conjugate. With $H_0 = I$ DFP becomes a conjugate gradient method. To afford some flexibility to the user MIN lets him choose a scale factor $\alpha$ and uses $H_0 = \alpha I$. It is suggested in the literature [6] that sensitivity of DFP to accuracy of line search may be reduced by restarting after every m iterations where $m \leq n$, i.e. by setting $H_m = H_0$. MIN allows the user to choose m.

The DFP method exhibits "poor convergence" characteristics if the Hessian of V0 has a poor eigenvalue structure. To remedy this Orien and Luenberger [7] have developed a self-scaling procedure which is incorporated in MIN as follows:

$$H_{k+1} = r_k \left[ H_k - \frac{H_k \, \Delta g_k \, \Delta g_k^T \, H_k}{\Delta g_k^T \, H_k \, \Delta g_k} \right] + \frac{p_k \, p_k^T}{p_k^T \, \Delta g_k}$$

$$r_k = \frac{p_k^T \, \Delta g_k}{\Delta g_k^T \, H_k \, \Delta g_k}$$

The user has the option of using self-scaling or not. These options make MIN very versatile and useful both for educational purposes and problem solving.

3.2.2 PARTAN: This method (of parallel tangents) was first developed by Shah et al. [8] based on geometric properties of the contours of a quadratic V0. It is a particular implementation of the method of conjugate gradients (see [5] p. 185).

The search directions $s_k$ are generated by two line searches as indicated in Fig. 9. In the first iteration $x_1$ is found from $x_0$ by a standard steepest descent step. After that $x_{k+1}$ is determined from $x_k$ by two line searches, i.e., $y_k$ is first found by a standard steepest descent step from $x_k$ and then $x_{k+1}$ is found by locating the minimum point on the line connecting $x_{k-1}$ and $y_k$. The process is continued for n steps and then restarted with a standard steepest descent step.

This method has strong global convergence characteristics. Also the line searches need not be as accurate as in other methods.

3.2.3 FR: The search direction algorithm FR was developed by Fletcher and Reeves [9]. This method is an implementation of the conjugate gradient algorithm for non-quadratic V0 without using its hessian. The algorithm is simply,

$$s_{k+1} = -g_{k+1} + \beta_k s_k \quad , \qquad s_0 = -g_0$$

$$\beta_k = \frac{g_{k+1}^T g_{k+1}}{g_k^T g_k}$$

where $x_{k+1}$ is found by line search along $s_k$ from $x_k$.

The algorithm must be restarted after every n iterations (i.e., $s_n$ must be reset to $-g_n$). This restarting is important for assuring global convergence of the method.

3.2.4 GRADIENT Method: This is one of the oldest and most widely known methods for minimizing a function. In this method the search direction is always the negative gradient direction, i.e.,

$$s_k = -g_k \quad .$$

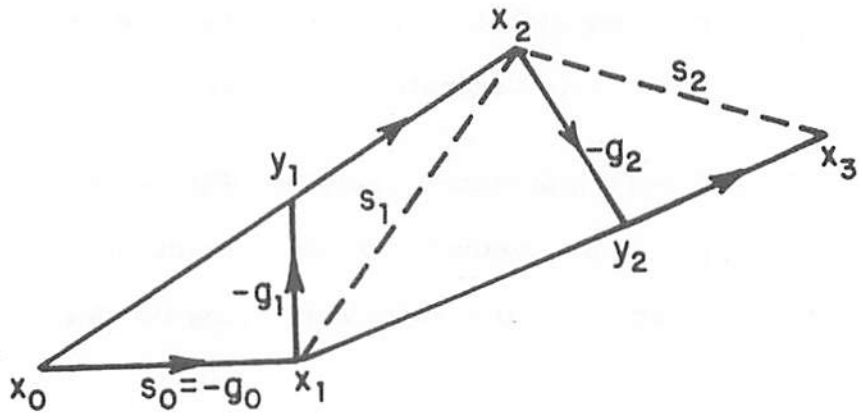


FIG. 9 ILLUSTRATING PARTAN

Most of other algorithms have been discovered by an attempt to modify this basic steepest descent method in such a way that the new algorithm will have superior convergence properties. This method has been included in MIN not only because it is the most often used technique but also because it continues to be the standard of reference against which other technics are measured.

### 3.3 Line Search Algorithms*

The basis of all the line search algorithms is to find an interval of acceptable length which contains the minimum of V0 in a particular direction as quickly as possible. That is they compute $\omega^*$ such that $V(\omega^*) \leq V(\omega)$ for all $\omega \geq 0$, where $V(\omega) \triangleq V0(x_k + \omega\, s_k)$. One method of computing $\omega^*$ is to determine an "uncertainty interval" UI = [a, b] such that $a \leq \omega^* \leq b$. The program stops when a small enough UI is found to meet the specified parameter tolerance. The property used to find UI is that if $V(\omega)$ is a unimodal function and $V(\omega_1) > V(\omega_3) < V(\omega_2)$ for some $\omega_1 < \omega_3 < \omega_2$ then $[\omega_1, \omega_2]$ is an UI. The triplet $\omega_1, \omega_2, \omega_3$ with above property is said to form a three-point pattern.

Each of the five line search algorithms (GS, FP, DSC, HO, FS) incorporated in MIN has the following common features:

(i) Start iteration at $\omega = 0$.

(ii) Find UI.

(iii) Find next iterate. Ensure new iterate lies in UI.

(iv) Use new iterate to reduce UI.

---

*The authors acknowledge with thanks the assistance of Rajan Suri in writing this section.

(v) If UI > parameter tolerance go to (iii).

However, the manner in which they implement (ii) and (iii) is different.

3.3.1 GS: The golden section algorithm is probably the best known [5, p. 134]. As stated earlier, the initial UI $[\omega_1, \omega_2]$ is found by locating $\omega_1 > \omega_3 > \omega_2$ such that $V(\omega_1) > V(\omega_3) < V(\omega_2)$. The implementation of GS in MIN is such that $\omega_3$ divides $[\omega_1, \omega_2]$ in the ratio G:1-G where $G = 1-G^2 \simeq .62$ is the golden ratio. The next iterate $\omega_4$ is chosen such that it divides $[\omega_1, \omega_2]$ in the ratio 1-G:G. This leads in step (iv) to a reduction in UI to G times the original length. Of the five line search algorithms the golden section proved to be the most robust.

3.3.2 FP: The false position algorithm attempts to find $\omega^*$ as a solution to the equation $g(\omega) \equiv (d/d\omega)\, V(\omega) = 0$ by a secant approximation. The iterations take the form,

$$\omega_{i+1} = \omega_i - g(\omega_i) \left[ \frac{\omega_i - \omega_{i-1}}{g(\omega_i) - g(\omega_{i-1})} \right]$$

Two safety features are incorporated in the FP implemented in MIN. They are,

i) Check for divergence caused by negative slope of $g(\omega)$, (see Figure 10).

ii) Check for slow convergence arising as shown in Figure 11. This is done by testing for an unbalanced* UI and choosing the next iterate to balance it.

---

*We call an UI $[\omega_1, \omega_3, \omega_2]$ unbalanced if one of the intervals $[\omega_1, \omega_3]$, $[\omega_3, \omega_2]$ is an order of magnitude larger than the other.

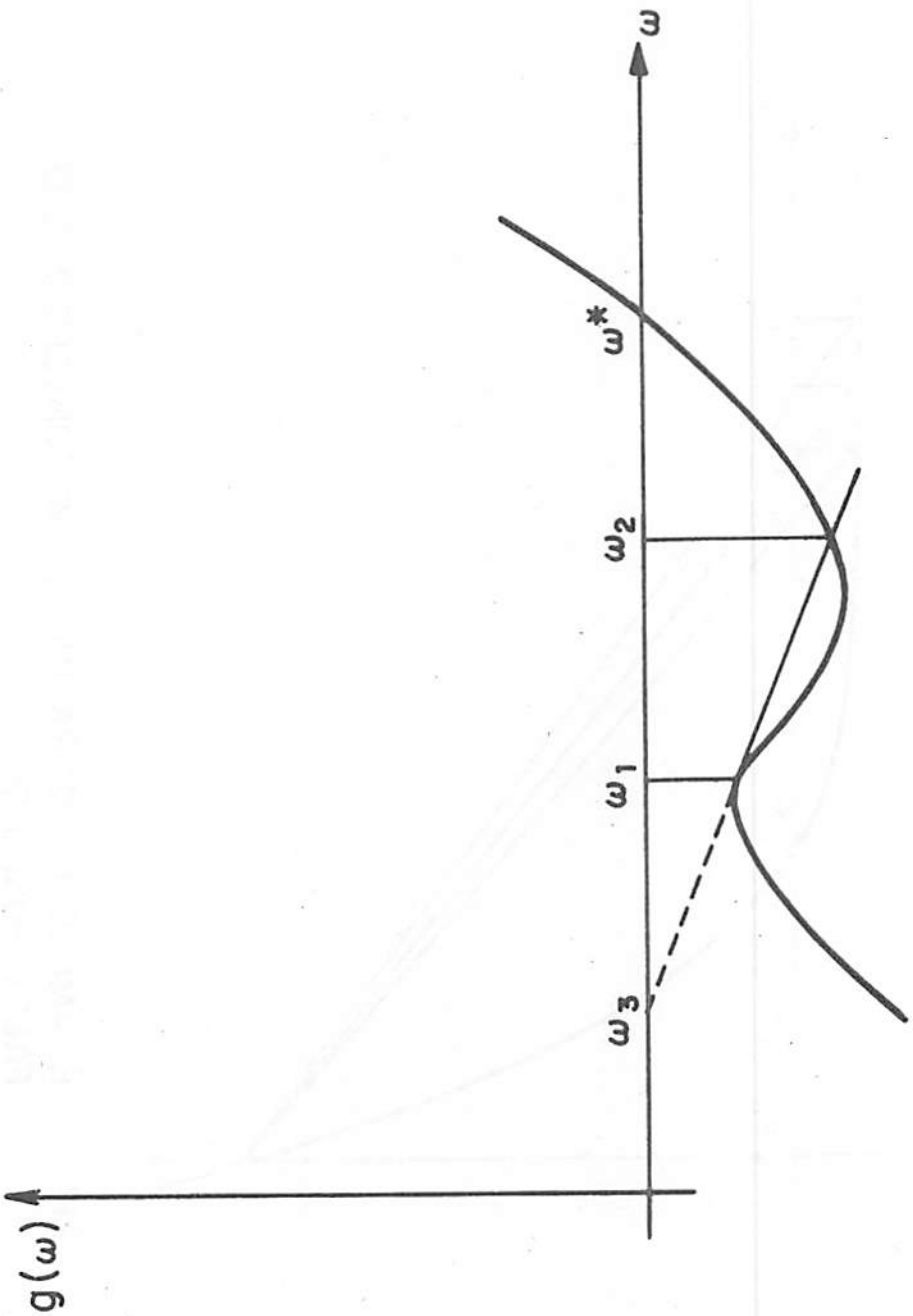


FIG. 10 DIVERGENCE OF FALSE POSITION DUE TO NEGATIVE SLOPE OF $g(\omega)$

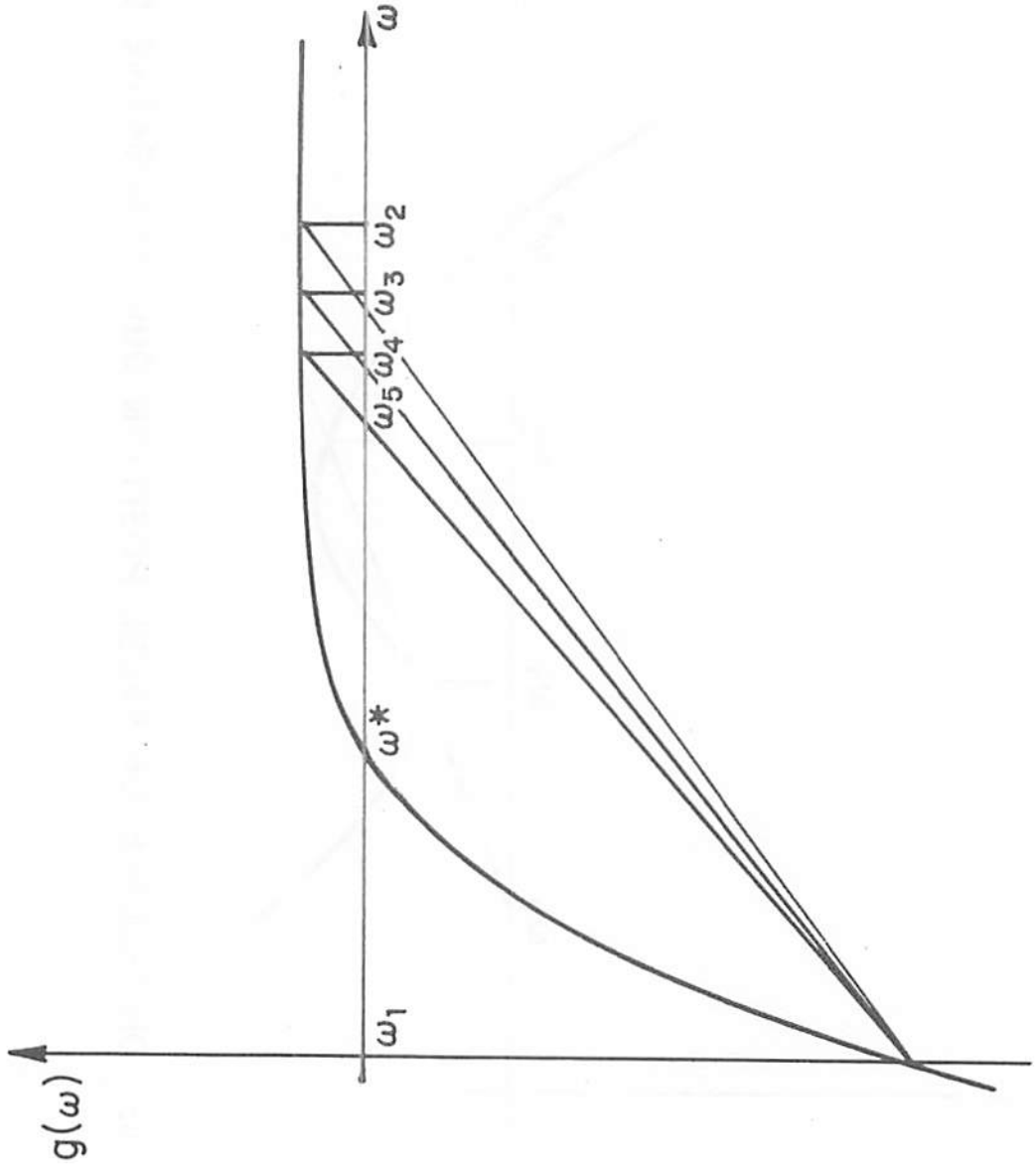


FIG. 11 EXAMPLE ILLUSTRATING SLOW CONVERGENCE OF FALSE POSITION

3.3.3 DSC-Powell [10, p. 44]: This has two stages. First, the algorithm of Davies, Swann, and Campey is used to find initial UI. This consists in doubling the step size until the minimum is overshot. The resulting three point pattern is refined by considering the mid-point of the last interval. This is followed by the Powell algorithm which uses a quadratic approximation on the three point pattern to estimate the minimum. The new point is used to form a three point pattern on a shorter UI. The Powell algorithm is repeated until required tolerances are met. The Powell algorithm is improved in MIN by testing for unbalanced UI and balancing it when necessary. This results in a more robust algorithm.

3.3.4 HO: The high-order method by Micchelli and Miranker [11] finds $\omega^*$ through the equation $g(\omega) = 0$. The basic idea here is to use bounds on $dg/d\omega$ to estimate the smallest UI as follows: Suppose $g_1 = g(\omega_1)$ and $g_2 = g(\omega_2)$ are found such that $g_1 \le 0 \le g_2$ and that $m \le dg/d\omega \le M$ in $[\omega_1, \omega_2]$. The new UI is taken to be $[\omega_1^*, \omega_2^*]$ where,

$$\omega_1^* = \begin{cases} \max\left(\omega_1 - \dfrac{g_1}{M},\ \omega_2 - \dfrac{g_2}{m}\right), & m > 0 \\ \omega_1 - \dfrac{g_1}{M}, & m \le 0 \end{cases}$$

$$\omega_2^* = \begin{cases} \min\left(\omega_1 - \dfrac{g_1}{m},\ \omega_2 - \dfrac{g_2}{M}\right), & m > 0 \\ \omega_2 - \dfrac{g_2}{M}, & m \le 0 \end{cases}$$

However, if m and M are not known then they are estimated as follows:

$$m_i = s\,\theta_i\,, \qquad M_i = s/\theta_i$$

$$\theta_0 = \frac{1}{2}\,, \qquad \theta_i = 1 - \frac{1}{2}(i - \theta_{i-1})$$

$$s = \frac{g_2 - g_1}{\omega_2 - \omega_1} \quad \text{at the i-th update.}$$

While estimating m and M as above a careful check is made by MIN to ensure that $\omega_1^*$ and $\omega_2^*$ are not chosen if they fall outside UI.

3.3.5 FS: The Fibonacci Search derives its popularity mainly because of its theoretical elegance. Given, an initial UI of length $\ell_0$ and a minimum acceptable separation $\epsilon$ for function evaluations this algorithm satisfies the specified parameter tolerance (i.e., the maximum acceptable length $\delta$ of the final UI) by using smallest number N of function evaluations. The procedure is as follows: (i) $I_0$, the initial UI has length $\ell_0$. Place the first function evaluation in $I_0$ at a distance $\ell_1$ from one end of $I_0$ where

$$\ell_1 = \frac{F_N}{F_{N+1}}\,(\ell_0 + F_{N-1}\,\epsilon) - F_{N-2}\,\epsilon$$

and $F_i$ are the Fibonacci numbers satisfying

$$F_i = F_{i-1} + F_{i-2}\,, \qquad F_0 = F_1 = 1$$

(ii) Reduce $I_0$ to $I_1$ by placing the second function evaluation at the reflection of the first about the mid-point of $I_0$.

(iii) $I_1$ will then contain a function evaluation.

(iv) Reduce $I_1$ to $I_2$ similar to step (ii) and repeat until $I_N$ has been found.

In practice, it was found that the number of function evaluations were about the same for FS and GS. Besides, this number was not at all sensitive to typical changes in $\epsilon$ (see Table 1). It is therefore felt that the added burden of computing the Fibonacci numbers is not worth the trouble.

### 3.4 Selecting Line Search Algorithms

A comparative study of the five line search algorithms was made on several test functions and the results in Table 2 are intended to help the inexperienced user in selecting the line search algorithm for his minimization program. The performance of the five line search algorithms on a particular test function, $V(\omega) = \omega + e^{1-\omega}$, are shown in Table 3. The reasons forselecting this test function are:

i) Transcendal functions are commonly used to test line search algorithms because no rational curve can fit them exactly.

ii) For $1 - \omega = \delta << 1$ we have $g(\omega) = 1 - e^{\delta} \simeq \delta$. Thus an uncertainty of $\pm \delta$ in $g(\omega)$ is approximately equivalent to a similar uncertainty in $\omega$ about the point $\omega = 1$ where $V(\omega)$ has its minimum. This permits fair comparison of line search algorithms involving gradient tolerances with those involving parameter tolerances.

TABLE 1

No. of function evaluations for Fibonacci search

| | $\varepsilon$ (Minimum acceptable separation) | | | | |
|---|---|---|---|---|---|
| Final UI | $10^{-3}$ | $10^{-4}$ | $10^{-6}$ | $10^{-9}$ | $10^{-\infty}$ |
| $10^{-1}$ | 10 | 10 | 10 | 10 | 10 |
| $10^{-3}$ | | 19 | 19 | 19 | 19 |
| $10^{-5}$ | | | 29 | 29 | 29 |
| $10^{-7}$ | | | | 38 | 38 |
| $10^{-9}$ | | | | | 48 |

TABLE 2

Comparison of line search algorithms

| Criterion for selection | Golden Sear. | False Psn. | DSC-Powell | High order |
|---|---|---|---|---|
| No information on function | G | B | N | B |
| No idea of initial step | G | | G | |
| Small number of fcn evals. | B | G | G | G |
| Speed of computation | G | N | B | G |
| High accuracy req'd. (Final UI < $10^{-5}$) | B (takes too long) | G | B (numerical errors) | G |
| Low accuracy rq'd. (Final UI > $10^{-5}$) | G | | G | |

NOTE: The Fibonacci Search has the same characteristics as the Golden Section Search, except that it would be optimum when the minimum separation is specified.

KEY:
G = Good
N = Neutral
B = Bad
Blank = Not sure

TABLE 3[+]

Number of function or gradient evaluations required for minimization of $V(\omega) = \omega + e^{1-\omega}$

| Tolerance on Gradient or Parameter | GS | FP | DSC | HO | FS |
|---|---|---|---|---|---|
| $10^{-1}$ | 11 | 8 | 7 | 3 | 10 |
| $10^{-3}$ | 20 | 9 | 9 | 9 | 19 |
| $10^{-5}$ | 30 | 10 | 10 | 11 | 29 |
| $10^{-7}$ | 40 | 14 | * | 11 | 38 |
| $10^{-9}$ | 49 | 15 | * | 13 | 48 |

[+]Initial UI for all algorithms was [0, 2.1]

*At points very close to the minimum, the DSC-Powell algorithm ran into difficulty due to the number of squaring and adding operations in the Powell algorithm, which led to larger numerical error than the UI.

# REFERENCES


1. WANG 2200 A/B Reference Manual, WANG Laboratories Inc., Mass., 1974.

2. Fletcher, R., "Optimization", Proc. of Symposium of the Institute of Mathematics and its Application, Academic Press, London, 1969.

3. Davidon, W.C., "Variable Metric Method for Minimization", Research and Development Report ANL-5990 (Ref.) U.S. Atomic Energy Commission, Argonne National Laboratories, 1959.

4. Fletcher, R. and Powell, M.J.D., "A Rapidly Convergent Descent Method for Minimization", Computer J., 6, 1963, pp. 163-168.

5. Luenberger, D.G., "Introduction to Linear and Nonlinear Programming", Addison Wesley, Mass. 1965.

6. Goldfarb, D., "Sufficient Conditions for the Convergence of a Variable Metric Algorithm", Chapter 18 in Ref. 2 above.

7. Crien, S.S. and Luenberger, D.G., "The Self-Scaling Variable Metric Algorithm (SSVM)", Fifth Hawaiian International Conference on Systems Sciences, January 1972.

8. Shah, B., Buehler, R., and Kempthorne, O., "Some Algorithms for Minimizing a Function of Several Variables", J. Soc. Indust. Appl. Math., 12, 1964, pp. 74-92.

9. Fletcher, R. and Reeves, C.M., "Function Minimization by Conjugate Gradients", Computer J. 7, 1964, pp. 149-154.

10. Himmelblau, D.M., "Applied Nonlinear Programming," McGraw-Hill, 1972.

11. Micchelli, C.A. and W.L. Miranker, "High Order Search Methods for Finding Roots", J.A.C.M., Vol. 22, No. 1, Jan. 1975, pp. 51-60.

# APPENDIX

## USER'S GUIDE TO MIN

The purpose of this appendix is to be a self contained guide to assist you in using MIN, assuming that you have a fair knowledge of WANG 2200 BASIC. A reading of earlier sections, though helpful in understanding MIN itself, is not necessary to use MIN.

MIN is an interactive computer program package designed as an educational tool to solve your unconstrained function minimization problem. Three essential features of any function minimization program are:

(i) a stopping criterion,

(ii) a minimization (search direction) algorithm, and

(iii) a line search algorithm.

MIN provides these features in the form of several ready made building blocks which you may use to build your own custom-made program. Currently, MIN (version 6) provides a choice of eight different combinations of three stopping criteria, four search direction algorithms and five line search algorithms. Thus, there is a choice of 160 ($= 8 \times 4 \times 5$) different configurations for your minimization program. What is more, you may switch building blocks to change configurations between iterations while solving a single problem.

This appendix is divided into five sections as follows:

A. 1: How to specify your problem

A. 2: How to LOAD and RUN MIN

A. 3: Special features of MIN

A. 4: How to have a dialog with MIN

A. 5: Sample dialog

It is suggested that you read Sections A. 1 - A. 3 before using MIN. Section A. 4 is best used as a guide on-line. To use it off-line it is best to simultaneously consult Section A. 5.

## A.1 HOW TO SPECIFY YOUR PROBLEM

If your function minimization problem is

$$\min_{x \in \Omega} \quad V0(x)$$

where the feasible set $\Omega = R^n$ (no constraints), $n \leq 6$ * and the cost function V0 has a derivative V1 then you may use MIN to solve your problem. To begin with, MIN expects you to supply the following BASIC statements:

1. Problem title as a DATA statement. First 30 characters of this title are used in intermediate and final result outputs.
2. Number of variables (dimension of x) as a DATA statement. This number must be less than or equal to 6.
3. A subroutine DEFFN'50 to define the function V0
4. A subroutine DEFFN'51 to define the gradient vector V1.

Extra care must be taken when specifying V0 and V1. This is because MIN does not check for errors in these. In particular, you should observe the following rules:

1. Do not use BASIC statement numbers <u>below</u> 8000
2. The two DATA statements should be strictly in the order shown.
3. The DEFFN' numbers 50 and 51 are reserved for V0 and V1, <u>respectively</u>. They should not be changed or interchanged.

---

* We limit $n \leq 6$ to permit use of MIN on a WANG 2200 with a modest memory size of 8k words.

4. In DEFFN'50 and DEFFN'51 use only the following variable names:

   V0 = Function value - a numeric scalar

   X(·) = Parameters - a numeric array

   V1(·) = Gradient vector - a numeric array

   Dimensions of X and V1 are predefined in MIN (by a DIM V1(6), X(6) statement) and you should not redefine (or define) their dimensions even if your problem dimension is different.

5. If you need additional variables (such as for storing intermediate calculations), use names Z1 through Z9 (scalar or array). No other names should be used. A DIMension statement must be included if you use any array variables other than V1 and X.

6. Before entering your problem make sure that the computer memory is clear. To do this, press CLEAR key followed by EXECUTE (CR/LF) key.

It is recommended that you save your problem specification on cassette for possible future use. This may be done as follows:

- Mount your cassette on tape drive 10B and rewind it
- Key in SAVE /10B, 8000 CR/LF
- Wait for problem to be saved on your cassette
- Rewind and remove your cassette

Example: Suppose you wish to minimize Rosenbrock's function:

$$\alpha(x_1-x_2^2)^2 + (1-x_1)^2 \qquad \text{for } \alpha = 1000 .$$

You may specify this problem as follows:

```
8000 DATA "ROSENBROCK'S FUNCTION-1000"
8010 DATA 2
9000 DEFFN'50:Z9=1000
9010 V0=Z9* (X(1)! 2-X(2))! 2+(1-X(1))! 2:RETURN
9500 DEFFN'51:V1(1)=4* X(1)* Z9* (X(1)! 2-X(2))-2*(1-X(1)):V1(2)=-2*
       Z9*(X(1)! 2-X(2)):RETURN
```

## A.2 HOW TO LOAD AND RUN MIN

The steps in using MIN are as follows:

1. Switch on the WANG 2200. Enter your problem specification and save it on your cassette as explained in Section A. 1.
2. Mount the MIN (version 6) cassette on tape drive 10A and rewind it. Press the LOAD key followed by the CR/LF key. Wait for loading to complete.
3. Press the RUN key followed by CR/LF
4. MIN will ask if you have stored (i.e., loaded into memory) your problem. Respond Y for Yes and N for No. If your answer is N, MIN will show you how to store your problem and then stop.* Store your problem as indicated and go back to Step 3.
5. MIN will continue the dialog from here which is self explanatory (for details see Section A. 4).

---

* If you get an error message †ERR xxx .... then go back to Section A. 1 and check your problem specification carefully.

## A.3 SPECIAL FEATURES OF MIN

Before proceeding to have a dialog with MIN you should note the following special features provided by MIN for your convenience:

1. On line guidance: After asking you any question MIN follows it by a hint to help you with your answers.

2. Built-in error check: MIN checks errors in your responses and will reject nonsense responses from you, again guiding you to the correct response.

3. Flexible modification of input data: Once all data is in, MIN will display it and will guide you in changing any data, if necessary.

4. Data printing: Once all data is in and you have confirmed to MIN that there are no more changes, then MIN gives you an option to have all the input data printed, thereby relieving you from the chore of remembering them.

5. Key-in facility: MIN goes to great lengths to accomodate your desire to change any input data. Suppose you confirmed the input data to be correct then MIN builds up your minimization program and executes it. If, after a few iterations of your program execution, you decide to change the input data, then you may indicate this to MIN by pressing Special Function key 15 (in the top row of WANG 2200 keyboard). MIN will halt execution of your program, and ask you whether you wish to change input data. If your response is N, MIN will forget the interruption and program execution will continue. If your response is Y, MIN will let your program

complete the current iteration and then go back to the input dialog with you to let you make appropriate changes.

6. Machine-state indication: To keep you informed of what is happening, MIN displays a message indicating current machine state. This message appears on the top left corner of the screen and may be one of the following: Waiting for Input, Loading, Executing, or Printing. In particular, if MIN indicates a state "Printing", make sure that the typewriter is switched ON and is in Auto mode so that it is ready for printing.

7. Continuation after stopping: When any of the chosen stopping criteria is satisfied, MIN stops executing your program and displays the "final results". At this point you may stop or continue minimization after making suitable changes in input data. For example, suppose you chose a stopping criterion of three iterations and the program stopped. Then you may switch the minimization algorithm and continue the iterations from the new point or you may change the starting point and continue, etc.

## A.4 HOW TO HAVE A DIALOG WITH MIN

This section is meant to be used as an on-line reference during dialog with MIN. In the discussion below, we reproduce the questions exactly as asked by MIN and follow it with explanations and helpful hints for your answer.

1. WHICH MINIMIZATION ALGORITHM # DO YOU WANT TO USE (1=DFP, 2=PARTAN, 3=FLETCHER REEVES, 4=GRADIENT)

   You have a choice of four different minimization algorithms:

   1. DFP (Davidon Fletcher Powell) method
   2. PARTAN (Parallel Tangent) method
   3. FLETCHER-REEVES (conjugate gradient) method
   4. GRADIENT (steepest descent) method

Input the number (1 through 4) of the algorithm that you want to use. The number of subsequent questions asked will depend on the method chosen. (For details of these four methods, see Section 3.2.) Since the relative performance of these methods depends on the function to be minimized, you are encouraged to try out as many of them as you can. From experience, for complicated functions, these methods generally rank in the order in which they have been listed above.

2. WHICH LINE SEARCH ALGORITHM # DO YOU WANT TO USE (1=GOLDEN SECTION, 2=FALSE POSITION, 3=DSC POWELL, 4=HIGH ORDER, 5=FIBONACCI)

You have a choice of five different line search algorithms as indicated. Enter the number (1 through 5) of the algorithm chosen by you. For details of these algorithms, see Section 3.3. There is no

definite order in which they are expected to perform. Table 2 (Section 3.4) summarizes the knowledge gained from our experience.

3. STARTING POINT
X( 1 )=

X( 2 )=

⋮

MIN asks you to give values for $x_1, x_2, x_3, \cdots$ successively. For example, if your problem dimension is three, you will be asked to specify X(1), X(2), and X(3).

4. GRADIENT TOLERANCE

This is the point where you choose one of the three stopping criteria provided by MIN. The minimization program will stop if gradient norm is reduced below this tolerance (usually 0 to $10^{-3}$). Since gradient norm can never be negative, a zero tolerance will nullify this criterion, i.e., the program will never stop by satisfying this criterion. Nullifying a stopping criterion normally results in the program coming to a stop on some other criterion after much longer execution time. It is therefore suggested that you start with a coarse tolerance (such as $10^{-4}$) and make it finer (e.g., $10^{-6}$) if program stops on previous tolerance. For more discussion on stopping criteria, see Section 3.1.

5. PARAMETER TOLERANCE

This is the second of the three stopping criteria provided by MIN. The program will stop whenever parameter improvement (i.e., step size $\|x_{k+1} - x_k\|$ ) in any iteration is below the tolerance you specify here. (For the significance of this criterion, see Section 3.1). You

are allowed to input any non-negative value (normal range 0 to $10^{-3}$). However, it is suggested that non-zero values less than $10^{-10}$ be not used. Otherwise, truncation errors may become comparable to tolerance and cause unpredicable problems. You may still input 0 to nullify this criterion so that the program will not check parameter improvement. In this case, MIN intelligently avoids truncation error problems by suitably adjusting the tolerance internally.

"Parameter tolerance" as used here should not be confused with "parameter accuracy". A parameter tolerance of $10^{-3}$ does not guarantee that final results will be accurate to 3rd decimal digit.

6. NUMBER OF ITERATIONS (I.E. LINE SEARCHES) REQUIRED*

This is the last of the three stopping criteria provided by MIN. The program will stop after executing the specified number of iterations. Any number from 1 through $10^{99}$ may be specified. 0 is not acceptable. A large number will essentially nullify this criterion as the program will invariably stop on some other criterion.

Note that you can nullify all the three stopping criteria. In this case, MIN will stop on one of two additional stopping criteria over which you have little control. These are:

1. Stop when MIN thinks that the minimum has been found
2. Stop when search direction finding algorithm fails and no restart is allowed by you.

---

* To avoid any ambiguity, we take one iteration to be synomymous with one line search.

7. HOW MUCH INTERMEDIATE RESULTS DO YOU WANT ON SCREEN (0=NIL, 1=ONLY FN. VALUE + NORM OF GRADIENT, 2=X, V0, V1, STEP SIZE)

You have three options for format of intermediate results on the screen:

1. No intermediate result
2. Function value and gradient norm listed in a tabular form (Fig. 12)
3. Parameter values, function value, gradient value, step size, etc. listed in detail as shown in Fig. 13.

If option 1 is chosen, the screen (except for machine state indication) will remain blank during execution. This choice may make you loose your patience, and does not result in any significant time saving because screen display is a very fast process.

8. HOW MUCH INTERMEDIATE RESULTS DO YOU WANT ON TYPEWRITER (0=NIL, 1=ONLY FN. VALUE +NORM OF GRADIENT, 2=X, V0, V1, STEP SIZE)

The same three options as in question 7 above are available. However, since it takes a long time to print the result on the typewriter, this choice should be made more carefully. You may choose any of the nine possible combinations of formats on the screen and typewriter. A good choice is to select option 3 (detailed results) for screen and option 2 (tabular results) for the typewriter.

9. HOW SHOULD THE FINAL RESULT BE OUTPUTED (1-DISPLAYED ON CRT, 2-PRINTED, 3-BOTH PRINTED AND DISPLAYED)

There is only one format of the final result. However, you may choose to display it on the screen and/or typewriter.

INTERMEDIATE RESULT FOR ROSENBROCK'S FUNCTION-100
(USING DFP WITH GOLDEN SECTION)

| ITERATION# | VALUE OF FUNCTION | NORM OF GRADIENT |
|---|---|---|
| 0 | 1 | 2 |
| 1 | .771109685344 | 5.2010728168 |
| 2 | .62369020096 | 7.5336017466 |
| 3 | .4364478606411 | 2.9676839743 |
| 4 | .317283836438 | 3.8135783301 |
| 5 | .274518135716 | 6.6445842391 |

FIGURE 12: Tabular Form for Intermediate Results (Option 2)

```
INTERMEDIATE RESULT FOR ROSENBROCK'S FUNCTION-100
(USING DFP WITH GOLDEN SECTION)
ITERATION #   6
X = ( .8720204592283 , .7557349692972 )
VALUE OF FUNCTION =  1.85734155E-02
NORM OF GRADIENT =  1.6664451496
LENGTH OF STEP =  .5932028764269

INTERMEDIATE RESULT FOR ROSENBROCK'S FUNCTION-100
(USING DFP WITH GOLDEN SECTION)
ITERATION #   7
X = ( .8686229159788 , .7531665451594 )
VALUE OF FUNCTION =  1.74392905E-02
NORM OF GRADIENT =  .33581373784
LENGTH OF STEP =  4.25911994E-03
```

FIGURE 13: Detailed Intermediate Results (Option 3)

10. # OF ITERATIONS FOR RESTART (RESETTING INVERSE HESSIAN)
(-1=RST ONLY IF NECESSARY, 0=NO RST, K=RST AFTER EVERY K ITERTN)

This and the following two questions will be asked only if you chose DFP method (in question 1). All search direction finding algorithms use negative gradient directions at the start and then use some formula to determine successive search directions. Some times the search direction algorithms give a direction along which there is no descent and thus fail. To avoid this situation, PARTAN and FLETCHER-REEVES algorithms specify that after n (= number of variables) iterations the program must be restarted, i.e., search direction must be reset to gradient direction. DFP does not have any rigid restarting rules and often one may get away without restarting.

Restarting option has been included to allow you to learn more about its effects by trying several different options. The available options are:

0: No restart (Program will stop if DFP fails)

K: Restart after every K iterations. Generally K is chosen to be equal to the number of variables

-1: Restart if method fails.

A message "RESTARTED" will appear for a brief moment at the top right corner of the screen during execution every time a restart occurs.

11. DO YOU WANT SELF SCALING
(Y OR N)

This feature is also exclusive to the DFP method. The convergence of DFP method depends on the inverse hessian of V0. A simple scaling (multiplying inverse hessian by a scalar) sometimes significantly improves

convergence. The self scaling method [7] incorporated in MIN scales the inverse hessian if you choose this option. This option is provided to allow you to compare the performance of DFP with and without self scaling. For details of self scaling, see Section 3. 2. 1.

12. INITIAL SCALE FACTOR (INVERSE HESSIAN WILL INITIALLY BE SET EQUAL TO THIS FACTOR TIMES IDENTITY MATRIX)

As discussed above, a simple scaling of inverse hessian sometimes results in significant improvement of convergence. Initially, hessian inverse is set equal to $\alpha I$ where $I$ is the identity metric and $\alpha$ is the initial scale factor you choose. The closer $\alpha I$ approximates the true inverse hessian the better is the convergence. If you do not have any idea of the inverse hessian at the starting point, $\alpha = 1$ is a safe choice.

## A.5 SAMPLE DIALOG AND SOLUTION

This section shows an actual RUN of MIN. The responses from the user to MIN are easily identified by the question mark (?) preceding those responses.

```
:RUN

HAVE YOU STORED FUNCTION ETC.
(Y OR N)
? Y

1.WHICH MINIMIZATION ALGORITHM # DO YOU WANT TO USE
(1=DFP,2=PARTAN,3=FLETCHER REEVES,4=GRADIENT)
? 1

2.WHICH LINE SEARCH ALGORITHM # DO YOU WANT TO USE
(1=GOLDEN SECTION,2=FALSE POSITION,3=DSC POWELL,
4=HIGH ORDER,5=FIBONACCI)
? 1

3.STARTING POINT
X( 1 )=
? Y
NUMERIC ANSWER (MAX 13 DIGITS ) REQUESTED. PLEASE INPUT AGAIN
? 0
X( 2 )=
? 0

4.GRADIENT TOLERANCE
? -.001
ANSWER MUST BE GREATER THAN OR EQUAL TO  0 . PLEASE INPUT AGAIN
? 1E-4

5.PARAMETER TOLERANCE
? 0

6.NUMBER OF ITERATIONS(I.E.  INE SEARCHES) REQUIRED
? 0
ANSWER MUST BE GREATER THAN OR EQUAL TO  1 . PLEASE INPUT AGAIN
? 5

7.HOW MUCH INTERMEDIATE RESULTS DO YOU WANT ON SCREEN.
(0=NIL,1=ONLY FN. VALUE+NORM OF GRADIENT,2=X,V0,V1,STEP SIZE)
? 0.5
ANSWER MUST BE INTEGER. PLEASE INPUT AGAIN
? 2

8.HOW MUCH INTERMEDIATE RESULTS DO YOU WANT ON TYPEWRITER.
(0=NIL,1=ONLY FN. VALUE+NORM OF GRADIENT,2=X,V0,V1,STEP SIZE)
? 1

9.HOW SHOULD THE FINAL RESULT BE OUTPUTED
(1-DISPLAYED ON CRT,2-PRINTED,3-BOTH PRINTED AND DISPLAYED)
? 3
```

```
10.# OF ITERATIONS FOR RESTART(RESETTING INVERSE HESSIAN)
(-1=RST ONLY IF NECESSARY,0=NO RST,K=RST AFTER EVERY K ITERTN)
? -1

11.DO YOU WANT SELF SCALING
(Y OR N)
? N

12.INITIAL SCALE FACTOR(INVERSE HESSIAN WILL INITIALY BE SET
EQUAL TO THIS FACTOR TIMES IDENTITY MATRIX)
? 1
HERE ARE THE CURRENT DATA FOR ROSENBROCK'S FUNCTION-100
1.MINIMZATION ALGORITHM - DFP
2.LINE SEARCH - GOLDEN SECTION
3.STARTING POINT X = ( 0 , 0 )
4.GRADIENT TOLERANCE =  1.000000000E-04
5.PARAMETER TOLERANCE =  0
6.NUMBER OF ITERATIONS REQUIRED =  5
7.HOW MUCH INTERMEDIATE RESULT ON SCREEN... 2
8.HOW MUCH INTERMEDIATE RESULT ON TYPEWRITER... 1
9.HOW SHOULD THE FINAL RESULT BE OUTPUTED... 3
10.RESTART OPTION...-1
11.SELF SCALING...N
12.INITIAL SCALE FACTOR =  1
PLEASE ENTER THE SERIAL # OF THE DATA TO BE CHANGED,IF ANY.
(0-FOR NO CHANGE)
? 2

2.WHICH LINE SEARCH ALGORITHM # DO YOU WANT TO USE
(1=GOLDEN SECTION,2=fa SE POSITION,3=DSC POWELL,
4=HIGH ORDER,5=FILONACCI)
? 2


HERE ARE THE CURRENT DATA FOR ROSENBROCK'S FUNCTION-100
1.MINIMZATION ALGORITHM - DFP
2.LINE SEARCH - FALSE POSITION
3.STARTING POINT X = ( 0 , 0 )
4.GRADIENT TOLERANCE =  1.000000000E-04
5.PARAMETER TOLERANCE =  0
6.NUMBER OF ITERATIONS REQUIRED =  5
7.HOW MUCH INTERMEDIATE RESULT ON SCREEN... 2
8.HOW MUCH INTERMEDIATE RESULT ON TYPEWRITER... 1
9.HOW SHOULD THE FINAL RESULT BE OUTPUTED... 3
10.RESTART OPTION...-1
11.SELF SCALING...N
12.INITIAL SCALE FACTOR =  1
PLEASE ENTER THE SERIAL # OF THE DATA TO BE CHANGED,IF ANY.
(0-FOR NO CHANGE)
? 0

  .
  .
  .
```

```
HERE ARE THE CURRENT DATA FOR ROSENBROCK'S FUNCTION-100
1.MINIMZATION ALGORITHM - DFP
2.LINE SEARCH - GOLDEN SECTION
3.STARTING POINT X = ( 0 , 0 )
4.GRADIENT TOLERANCE =  0
5.PARAMETER TOLERANCE =  0
6.NUMBER OF ITERATIONS REQUIRED =  1000
7.HOW MUCH INTERMEDIATE RESULT ON SCREEN... 2
8.HOW MUCH INTERMEDIATE RESULT ON TYPEWRITER... 1
9.HOW SHOULD THE FINAL RESULT BE OUTPUTED... 3
10.RESTART OPTION...-1
11.SELF SCALING...N
12.INITIAL SCALE FACTOR =  1
```

INTERMEDIATE RESULT FOR ROSENBROCK'S FUNCTION-100
(USING DFP WITH GOLDEN SECTION)

| ITERATION# | VALUE OF FUNCTION | NORM OF GRADIENT |
|---|---|---|
| 0 | 1 | 2 |
| 1 | .771109685344 | 5.2010728168 |
| 2 | .62369020096 | 7.5336017466 |
| 3 | .4364478606411 | 2.9676839743 |
| 4 | .3172838636438 | 3.8135783301 |
| 5 | .2745181357166 | 6.6445842391 |
| 6 | 1.85734155E-02 | 1.6664451496 |
| 7 | 1.74392905E-02 | .33581373784 |
| 8 | 6.12664800E-03 | 2.0063248797 |
| 9 | 1.17954014E-03 | .76001095176 |
| 10 | 1.42088875E-04 | .30090057751 |
| 11 | 4.65565508E-06 | 9.12062855E-02 |
| 12 | 6.08894230E-09 | 1.30163586E-03 |
| 13 | 1.15552929E-12 | 4.12928664E-05 |
| 14 | 3.58238844E-15 | 2.25641321E-06 |
| 15 | 1.80000000E-19 | 1.39556440E-08 |
| 16 | 2.42500000E-21 | 1.96606815E-09 |
| 17 | 1.01000000E-22 | 4.45425639E-10 |
| 18 | 1.01000000E-22 | 4.45425639E-10 |

```
FINAL RESULT FOR ROSENBROCK'S FUNCTION-100
(USING DFP WITH GOLDEN SECTION)
ITERATION #  19
X = ( 1.000000000001 , 1.000000000001 )
VALUE OF FUNCTION =  1.01000000E-22
NORM OF GRADIENT =  4.45425639E-10
LENGTH OF STEP =  0
# OF FUNCTION EVALUATIONS= 628
# OF GRADIANT EVALUATIONS= 36
STOPPING CRITERIA -
NOT MUCH DESCENT EVEN ALONG GRADIENT DIRECTION
(PROBABLY REACHED THE MINIMUM)
```

JOINT SERVICES ELECTRONICS PROGRAM

REPORTS DISTRIBUTION LIST

Chief, R and D Division
Defense Communications Agency
Washington, DC 20301

Defense Documentation Center
Cameron Station
Alexandria, Virginia 22314
ATTN: DDC-TCA (Mrs. V. Caponio) (12)

Dr. A.D. Schnitzler
Institute for Defense Analyses
Science and Technology Division
400 Army-Navy Drive
Arlington, Virginia 22202

Dr. George H. Heilmeier
Office of Director of Defense Research
and Engineering
The Pentagon
Washington, DC 20315

Director, National Security Agency
Fort George G. Meade, Maryland 20755
ATTN: Dr. T.J. Beahn

HQ/USAF (AF/RDPE)
Washington, DC 20330

HQ/USAF/RDPS
Washington, DC 20330

Rome Air Force Center
Griffiss AFB, New York 13440
ATTN: Documents Library (TILD)

Mr. H.E. Webb, Jr. (ISCP)
Rome Air Development Center
Griffiss AFB, New York 13440

AFSC (CCJ/Mr. Irving R. Mirman)
Andrews AFB
Washington, DC 20334

Directorate of Electronics and Weapons
HQ AFSC/DLC
Andrews AFB, Maryland 20334

Directorate of Science
HQ AFSC/DLS
Andrews AFB, Washington, DC 20331

Mr. Carl Sletten
AFCRL/LZ
L.G. Hanscom Field
Bedford, Mass. 01730

Dr. Richard Pitcard
AFCRL/OP
L.G. Hanscom Field
Bedford, Mass. 01730

LTC J.W. Gregory
AF Member, TAC
Air Force Office of Scientific Research
1400 Wilson Blvd.
Arlington, Virginia 22209 (5)

Mr. Robert Barrett
AFCRL/LQ
L.G. Hanscom Field
Bedford, Mass. 01730

Dr. John N. Howard
AFCRL (CA)
L.G. Hanscom Field
Bedford, Mass. 01730

HQ ESD (DRI/Stop 22)
L.G. Hanscom Field
Bedford, Mass. 01730

Prof. R.E. Fontana
Head Dept. of Electrical Engineering
AFIT/ENE
Wright-Patterson AFB, Ohio 45433

AFAL/TE, Dr. W.C. Eppers, Jr.
Chief, Electronics Technology Division
Air Force Avionics Laboratory
Wright-Patterson AFB, Ohio 45433

AF Avionics Lab/CA
Acting Chief Scientist
AF Avionics Laboratory
Wright-Patterson AFB, Ohio 45433
ATTN: Dr. Robert J. Doran

AFAL/TEA (Mr. R.D. Larson)
Wright-Patterson AFB, Ohio 45433

Faculty Secretariat (DFSS)
US Air Force Academy
Colorado 80840

Howard H. Steenbergen
Chief, Microelectronics Development and
Utilization Group/TE
Air Force Avionics Laboratory
Wright-Patterson AFB, Ohio 45433

Dr. Richard B. Mack - Physicist
Radiation and Reflection Branch (LZR)
Air Force Cambridge Research Labs.
L. G. Hanscom Field,
Bedford, Mass. 01730

Charles S. Sahagian
Chief, Preparation and Growth Branch (LQ)
Air Force Cambridge Research Labs.
L. G. Hanscom Field
Bedford, Mass. 01730

Major William Patterson
Assistant Chief, Information Processing
Branch (IS1)
Rome Air Development Center
Griffiss AFB, New York 13441

Major Richard J. Gowen
Tenure Professor
Dept. of Electrical Engineering
USAF Academy, Colorado 80840

Director, USAF Project RAND
Via: Air Force Liaison Office
The RAND Corporation
1700 Main Street
Santa Monica, California 90406
ATTN: Library D

AUL/LSE - 9663
Maxwell AFB, Alabama 36112

AFETR Technical Library
P.O. Box 4608, MU 5650
Patrick AFB, Florida 32925

ADTC (SSLT)
Eglin AFB, Florida 32542

HQ AMD (RDR/Col. Goddas)
Brooks AFB, Texas 78235

USAFSAM (RAT)
Brooks AFB, Texas 78235

Commander
White Sands Missile Range
New Mexico 88002
ATTN: STEWS-AD-L, Technical Library (2)

USAF European Office of Aerospace Research
Technical Information Office
Box 14, FPO New York 09510

VELA Seismological Center
312 Montgomery Street
Alexandria, Virginia 22314

Dr. Carl E. Baum
AFWL (ES)
Kirtland AFB, New Mexico 87117

Hqs. Elct. Sys. Division (AFSC)
L.G. Hanscom Field
Bedford, Mass. 01730
ATTN: ESD/MCIT/Stop 36
Mr. John Mott/Smith

USAFSAM/RAL
Brooks AFB, Texas 78235

Dr. Paul M. Kalaghan
AFCRL/LZN
L.G. Hanscom Field
Bedford, Mass. 01730

HQDA (DARD-ARS-P)
Washington, DC 20310

Commander, US Army Security Agency
Arlington Hall Station
Arlington, Virginia 22212
ATTN: IARD-T

HQ Army Materiel Command
Technical Library Rm 7S 35
5001 Eisenhower Avenue
Alexandria, Virginia 22304

Mr. H.T. Darracott (AMCAM-TF)
US Army Advanced Materiel Concepts Agency
5001 Eisenhower Avenue
Alexandria, Virginia 22304

Commander (AMXRD-BAD)
US Army Ballistics Research Laboratory
Aberdeen Proving Ground
Aberdeen, Maryland 21005

Commander
Picatinny Arsenal
Dover, N.J. 07701
ATTN: Science and Tech. Info. Br.
SMUPA-TS-T-S

Dr. Hermann Robl, Chief Scientist
US Army Research Office
Box CM, Duke Station
Durham, North Carolina 27706

Richard O. Ulsh (CRDARD-IP)
US Army Research Office
Box CM, Duke Station
Durham, North Carolina 27706

Mr. George C. White, Jr.
Deputy Director, L1000, 54-4
Pitman-Dunn Laboratory
Frankford Arsenal
Philadelphia, Penna. 19137

Redstone Scientific Information Center
US Army Missile Command
Redstone Arsenal, Alabama 35809
ATTN: Chief, Document Section

Commander
US Army Missile Command
Redstone Arsenal, Alabama 35809
ATTN: AMSMI-RR

Dr. Homer F. Priest
Chief, Materials Sciences Division, Bldg. 292
Army Materials and Mechanics Research Center
Watertown, Mass. 02172

John E. Rosenberg
Harry Diamond Laboratories
Connecticut Ave. and Van Ness St. N.W.
Washington, DC 20438

Commandant
US Army Air Defense School
Fort Bliss, Texas 79916
ATTN: ATSAD-T-CSM

Commandant
US Army Command and General Staff College
Fort Leavenworth, Kansas 66027
ATTN: Acquisitions, Lib. Div.

Dr. Hans K. Ziegler (AMSEL-TL-D)
Army Member, TAC/JSEP
US Army Electronics Command
Fort Monmouth, New Jersey 07703

Mr. I.A. Balton, (AMSEL-TL-DC)
Executive Secretary, TAC/JSEP
US Army Electronics Command
Fort Monmouth, New Jersey 07703 (5)

Mr. A.D. Bedrosian, Rm 26-131
US Army Scientific Liaison Office
M.I.T.
77 Massachusetts Avenue
Cambridge, Mass. 02139

Director (NV-D)
Night Vision Laboratory, USAECOM
Fort Belvoir, Virginia 22060

Commander/Director
Atmospheric Sciences Laboratory
White Sands Missile Range,
New Mexico 88002
ATTN: AMSEL-BL-DD

Atmospheric Sciences Laboratory
US Army Electronics Command
White Sands Missile Range
New Mexico 88002
ATTN: AMSEL-BL-RA (Dr. Holt)

Chief, Missile EW Tech. Area
Electronic Warfare Laboratory, ECOM
White Sands Missile Range,
New Mexico 88002
ATTN: AMSEL-WL-MY

US Army Armaments
Rock Island, Illinois 61201
ATTN: AMSAR-RD

US Army ABMDA
1300 Wilson Blvd
Arlington, Virginia 22209
ATTN: BDMD-NC, Mr. Gold

Harry C. Holloway, MD Col, MC
Director, Div. of Neuropsychiatry
Walter Reed Army Institute of Research
Washington, DC 20012

Commander, USASATCOM
AMCPM-SC
Fort Monmouth, New Jersey 07703

Director, TRI-TAC
Fort Monmouth, New Jersey 07703
ATTN: TT-AD (Mrs. Briller)

Commander
US Army R and D Group (Far East)
APO, San Francisco, California 96343

Commander, US Army Communications
Command
Fort Huachuca, Arizona 85613
ATTN: Director, Advanced Concepts Office

Project Manager, ARTADS
(AMCPM-TDS)
EAI Building
West Long Branch, New Jersey

US Army White Sands Missile Range
STEWS-ID-R
White Sands Missile Range, New Mexico 88002
ATTN: Dr. Alton L. Gilbert

Mr. William T. Kawal
US Army R and D Group (Far East)
APO, San Francisco, California 96343

Commander
US Army Electronics Command
Fort Monmouth, New Jersey 07703

ATTN: AMSEL-RD-O (Dr. W.S. McAfee)
CT-L (Dr. G. Bosser)
CT-LE (Dr. S. Kpetein)
BL-FM-A
CT-D
CT-R
NL-O (Dr. H.S. Bennett)
NL-T (Mr. R. Kulicke)
NL-C
NL-PB
NL-F
NL-M
TL-I
TL-B
VL-D
WL-D
TL-MM (Mr. Lipeta)
BL-FM (Mr. Edward Collett)
NL-O
NL-N
NL-M (Schwering)
NL-T
TL-DR
TL-E (Dr. S. Kronenberg)
TL-E (Dr. J. Kohn)
TL-I (Dr. C. Thornton)
NL-B (Dr. S. Amoroso)

Director, Electronic Programs
Office of Naval Research
800 North Quincy Street
Arlington, Virginia 22217
ATTN: Code 427

Director, Naval Research Laboratory
Washington, DC 20390
ATTN: Mr. A. Brodsinsky, Code 5200

Director, Naval Research Laboratory
Washington, DC 20390
ATTN: Library, Code 2629 (ONRL)

Dr. G. M. R. Winkler
Director, Time Service Division
US Naval Observatory
Washington, DC 20390

Naval Weapons Center
Technical Library (Code 753)
China Lake, California 93555

Director
Information Systems Program (437)
Office of Naval Research
Arlington, Virginia 22217

Director
Naval Research Laboratory (Code 6400)
4555 Overlook Avenue, SW
Washington, DC 20375

Director
Naval Research Laboratory (Code 6470)
4555 Overlook Avenue, SW
Washington, DC 20375

Director
Naval Research Laboratory (Code 6460)
4555 Overlook Avenue, SW
Washington, DC 20375

Dr. Leo Young (Code 5203)
Electronics Division
Naval Research Laboratory
Washington, DC 20375

Commander
Naval Training Equipment Center
Orlando, Florida 32813

Dr. A. L. Slafkosky
Scientific Advisor, Code AX
Hqs. US Marine Corps
Washington, DC 20380

US Naval Weapons Laboratory
Dahlgren, Virginia 22448

Commander
US Naval Ordnance Laboratory
Silver Spring, Maryland 20910
ATTN: Tech Library and Info Services Div

Director, Office of Naval Research
Boston Branch
495 Summer Street
Boston, Massachusetts 02210

Commander, Naval Missile Center
Point Mugu, California 93042
ATTN: 5632.2, Technical Library

Commander
Naval Electronics Laboratory Center
San Diego, California 92152
ATTN: Library

Deputy Director and Chief Scientist
Office of Naval Research Branch Office
1030 East Green Street
Pasadena, California 91106

Superintendent
Naval Post Graduate School
Monterey, California 93940
ATTN: Library (Code 2124)

Officer in Charge, New London Lab.
Naval Underwater Systems Center
(Tech. Library)
New London, Connecticut 06320

Commander, Naval Avionics Facility
Indianapolis, Indiana 46241
ATTN: D/035 Technical Library

Commander
Office of Naval Research Branch Office
536 South Clark Street
Chicago, Illinois 60605

Naval Air Development Center
Johnsville
Warminster, Pennsylvania 18974
ATTN: Technical Library

Naval Oceanographic Office
Technical Library (Code 1640)
Suitland, Maryland 20390

Naval Ship Research and Development Center
Central Library (Code L42 and L43)
Washington, DC 20007

Mr. F. C. Schwenk, RD-T
National Aeronautics and Space Administration
Washington, DC 20546

Los Alamos Scientific Laboratory
PO Box 1663
Los Alamos, New Mexico 87544
ATTN: Reports Library

M. Zane Thornton
Deputy Director Institute for Computer
Sciences and Technology
National Bureau of Standards
Washington, DC 20234

Director, Office of Postal Technology
(R and D)
US Postal Service
11711 Parklawn Drive
Rockville, Maryland 20852

NASA Lewis Research Center
21000 Brookpark Road
Cleveland, Ohio 44135
ATTN: Library

Library -R51
Bureau of Standards
Acquisition
Boulder, Colorado 80302

MIT Lincoln Laboratory
PO Box 73
Lexington, Mass. 02173
ATTN: Library A-082

Director
Research Laboratory of Electronics
MIT
Cambridge, Mass. 02139

Director, Microwave Research Institute
Polytechnic Institute of New York
Long Island Graduate Center, Route 110
Farmingdale, New York 11735

Mr. Jerome Fox, Research Coordinator
Polytechnic Institute of New York
333 Jay Street
Brooklyn, New York 11201

Director, Columbia Radiation Laboratory
Dept. of Physics
Columbia University
538 West 120th Street
New York, New York 10027

Director, Coordinated Science Laboratory
University of Illinois
Urbana, Illinois 61801

Director, Stanford Electronics Laboratory
Stanford University
Stanford, California 94305

Director, Microwave Laboratory
Stanford University
Stanford, California 94305

Director, Electronics Research Laboratory
University of California
Berkeley, California 94720

Director, Electronics Sciences Laboratory
University of Southern California
Los Angeles, California 90007

Director, Electronics Research Center
The University of Texas at Austin
Engineering-Science Bldg 112
Austin, Texas

Director of Laboratories
Division of Engineering and Applied Physics
Pierce Hall
Harvard University
Cambridge, Mass. 02138