NASA TECHNICAL NOTE

NASA TN D-5595

c. 1

NASA TN D-5595

# AUTOMATIC ANALOG COMPUTER SCALING USING DIGITAL OPTIMIZATION TECHNIQUES

*by John Celmer and Mary Rouland*

*Goddard Space Flight Center*
*Greenbelt, Md.*

NATIONAL AERONAUTICS AND SPACE ADMINISTRATION • WASHINGTON, D. C. • MARCH 1970

| | | |
|---|---|---|
| 1. Report No. NASA TN D-5595 | 2. Government Accession No. | 3. Recipient's Catalog No. |
| 4. Title and Subtitle Automatic Analog Computer Scaling Using Digital Optimization Techniques | | 5. Report Date March 1970 |
| | | 6. Performing Organization Code |
| 7. Author(s) John Celmer and Mary Rouland | | 8. Performing Organization Report No. G-950 |
| 9. Performing Organization Name and Address Goddard Space Flight Center Greenbelt, Maryland 20771 | | 10. Work Unit No. 604-31-03-12-51 |
| | | 11. Contract or Grant No. |
| 12. Sponsoring Agency Name and Address National Aeronautics and Space Administration Washington, D. C., 20546 | | 13. Type of Report and Period Covered Technical Note |
| | | 14. Sponsoring Agency Code |
| 15. Supplementary Notes | | |

16. Abstract

The Automatic Scaling Program provides optimum scaling of any number of differential equations for an analog simulation. A set of equations is defined to have optimum analog scaling when the deviation from unity of the gain at each amplifier is minimized. Striving to satisfy this criterion, the program evenly distributes the gains as close to unity as possible, as limited by the coefficient matrix of the specific system of equations. Both linear and nonlinear systems are scaled with only a simple numerical input. Constraints may be placed on the maximum value of any equation or all the maximums may be considered arbitrary; the former procedure produces optimum scaling subject to constraints and problem coefficients, and the latter results in optimum scaling subject only to problem coefficients. The output provides a documentation of the maximum, the level, and the potentiometer readings and gains.

| | |
|---|---|
| 17. Key Words Suggested by Author | 18. *Distribution Statement* Unclassified—Unlimited |

| | | | |
|---|---|---|---|
| 19. Security Classif. (of this report) | 20. Security Classif. (of this page) | 21. No. of Pages | 22. Price* |
| Unclassified | Unclassified | 30 | $3.00 |

*For sale by the Clearinghouse for Federal Scientific and Technical Information
Springfield, Virginia 32151

## CONTENTS

# AUTOMATIC ANALOG COMPUTER SCALING USING DIGITAL OPTIMIZATION TECHNIQUES

by
John Celmer and Mary Rouland*
*Goddard Space Flight Center*

## INTRODUCTION

The Automatic Scaling Program is a digital program designed to eliminate the tedious, time-consuming process of manually scaling a linear or nonlinear system of differential equations for an analog computer. Optimum scaling performed by the program minimizes the deviation from unity of each amplifier gain. Since the analog's accuracy is limited to four significant figures, any scaling that results in excessively high or low gains, that is, with a large deviation from unity, reduces precision even further. While it is important to recognize the program's time-saving feature and optimization of the analog's limited precision, it is even more important to emphasize that true optimum scaling is extremely difficult if not impossible to perform manually. All the inputs required are simple numerical data; the outputs include documentation of all maximums, levels, and potentiometer readings and gains.

## GENERAL PRINCIPLES

The Automatic Scaling Program is a digital program which optimizes the scaling of an arbitrary number of differential equations, linear or nonlinear, for an analog simulation. The obvious advantage of having a digital computer perform the scaling is the time saved for the engineer who formerly performed this tedious task manually. The primary advantage, however, is that the program produces an optimum set of maximum values.

The necessary input, designed for simplicity, requires only simple numerical data. The number of equations, NEQ, the coefficient matrix, A(I, J), and an integer matrix M(I, J) which identifies the variable or variables of each term, suffice to completely describe a system to be scaled. The input also allows the engineer to specify some constant maximum values or to place minimum and/or maximum constraints on one or more maximum. If only NEQ, A(I, J), and M(I, J) are used as input, that is, if no maximums are to be held constant or constrained, the program assumes the

*Stabilization and Control Branch Systems Division.

maximum value vector

$$\mathrm{AMA(I)} = 10.0 \qquad \mathrm{I} = 1, 2, \cdots, \mathrm{NEQ}$$

as a starting point for the search for the optimum vector.

Optimum scaling minimizes the deviation from unity of the gain at each amplifier. This gain, represented by a matrix, PG(I, J), describes the potentiometer gain of every term in all equations. Each element of the PG matrix

$$\mathrm{PG(I, J)} = \frac{[\mathrm{A(I, J)}]\left\{\mathrm{AMA}[\mathrm{M(I, J)}]\right\}}{\mathrm{AMA(I)}}$$

is calculated to minimize the merit value

$$\mathrm{MERIT} = \sum_{\mathrm{I}=1}^{\mathrm{NEQ}} \sum_{\mathrm{J}=1}^{\text{No. of terms}} [\mathrm{PG(I, J)} - 1]^2$$

where PG(I, J) is replaced by 1/PG(I, J) if PG(I, J) $<$ 1.0 and PG(I, J) is replaced by 1 if M(I, J) = I, in order that unchangeable first-order loop gains do not contribute to the merit value.

The AMA vector associated with the minimum merit value of a set of equations, located by the program's search routine running completely free without constant maximums or constraints as input, produces optimum scaling of the problem, that is, scaling that is subject only to problem coefficients. If some maximums are held constant or constrained, the scaling is optimum but it is subject to constraints and constant maximums as well as problem coefficients.

The principal method used in obtaining the minimum merit value is the gradient or direction of steepest descent in conjunction with the Fibonacci search procedure—a one-dimensional, sequential routine used to determine the optimum distance to move in the direction of steepest descent. In general, the slope or sensitivity of each maximum increases as its magnitude decreases, with very small maximums having very steep slopes. Therefore, if a problem has any maximums less than unity with steep slopes, the step size or distance moved in the steepest-descent direction is limited and the steepest-descent routine becomes ineffective in its convergence to the minimum merit value. The program follows a new search technique in which a modified steepest-descent movement is performed in three steps. With all AMA's less than or equal to unity held constant, the first step allows maximums greater than unity to move in the modified steepest descent direction until the merit value is minimized. The second step allows a similar movement for maximums greater than one-tenth but less than or equal to one, and the third step follows the same procedure for maximums less than or equal to one-tenth.

The three-step modified steepest-descent method continues its normal sequence unless the auxiliary search routine is activated by the manual turning on of a predesignated sense switch on the computer console. The auxiliary search is a combination sequential, linear interpolation routine in which the slope of each maximum less than one is individually minimized with respect to all other AMA's held constant. While this would be too time-consuming to be a part of the normal search procedure, it is useful as a "kicking" device near the beginning of the search or whenever the three-step modified steepest descent may be moving very slowly. If the search is actually near the true global minimum, the rate of change of the merit value due to the auxiliary search will not be much improved over the normal routine and will be much more time-consuming. However, if the modified steepest-descent search is slowed by a local minimum, the auxiliary search will "kick" it out of the region and allow the normal search to become effective again. Therefore, one of the two methods—the modified steepest-descent or the manually controlled auxiliary routine—will continue the search for the AMA vector that minimizes the merit value, until the precision or stepping condition specified by the program is reached.

All data used as input in the program, including any optional data, are printed for future reference and documentation. The merit value is printed at the conclusion of each search step in order that the programmer may follow the progress of the search and, with this information, decide when the auxiliary search might be useful. After optimum scaling is determined, documentation of results is provided. This includes the equation number, the maximum value, the level, the potentiometer readings and gains of each equation, and the minimum merit value. An additional feature in which all levels are rounded to one significant figure is also included as output, since convenient levels may be preferred. The merit value resulting from the rounding of levels is also given, to simplify evaluation of the rounding effect.

## STRUCTURE

### Required Input

The standard notation for systems of algebraic equations is

$$X(I) = \sum_{J} C(I, J) * X(J) ,$$

$$(I, J = 1, 2 \cdots N) ,$$

where $C(I, J)$ is an $N \times N$ coefficient matrix. For linear differential equations the standard notation is

$$\dot{X}(I) = \sum_{J} C(I, J) * X(J) ,$$

$$(I, J = 1, 2 \cdots N) .$$

Since analog simulations involve both summations and integrations it is convenient to mix both algebraic and differential equations

$$X(I) = \sum_K C(I, K) * X(K) ,$$

$$\dot{X}(J) = \sum_K C(J, K) * X(K) ,$$

$$(I = 1 \cdots NA;\ J = NA + 1 \cdots N) ,$$

$$(K = 1 \cdots N) .$$

The $N \times N$ matrix C frequently contains a profusion of zero entries (the number of nonzero entries in any row is usually less than five). It is thus more convenient to use an alternate notation involving an $N \times 4$ coefficient matrix A in conjunction with an $N \times 4$ integer matrix M

$$X(I) = \sum_K A(I, K) * X[M(I, K)] ,$$

$$\dot{X}(J) = \sum_K A(J, K) * X[M(J, K)] ,$$

$$(I = 1 \cdots NA;\ J = NA + 1 \cdots N) ,$$

$$(K = 1 \cdots 4) .$$

This approach reduces storage requirements and simplifies input preparation.

Assume the following set of differential equations:

$$\dot{X}(1) = -0.5\,[X(1)] + 0.09\,[X(2)] - 2.0\,[X(3) * X(1)] ,$$

$$\dot{X}(2) = +1.0\,[X(1)] - 2.0\,[X(2)] ,$$

$$\dot{X}(3) = -0.055\,[X(1) * X(2)] - 0.8\,[X(3)] .$$

The input required to scale this problem is NEQ (the number of equations), an A(I, J) matrix (coefficients of each term), and an M(I, J) matrix (the variable or variables of each term), where

I = 1, 2,· · ·, NEQ and J = 1, 2,· · ·, number of terms of (I).

NEQ = 3

A(I, J) Matrix

| | 1 | 2 | 3 |
|---|---|---|---|
| 1 | -0.5 | +0.09 | -2.0 |
| 2 | +1.0 | -2.0 | |
| 3 | -0.055 | -0.8 | |

M(I, J) Matrix

| | 1 | 2 | 3 |
|---|---|---|---|
| 1 | 1 | 2 | 3001* |
| 2 | 1 | 2 | |
| 3 | 1002* | 3 | |

Note: (*) indicates the form in which multipliers are entered as data. The format in which the program accepts these data will be included under "Operating Instructions—Required Input."

## Optional Input

The engineer may provide an initial set of maximums describing the point in space where the search will begin and/or he may specify that some of the maximums are to be held at constant values. The maximum value of any equation not included in the optional input is set to 10.0 before the search is begin. If the system is nonlinear, the engineer may want to place minimum and/or maximum constraints on the maximums of some equations. This is also possible in the optional input. Format for the four variations of optional input is found under "Operating Instructions–Optional Input."

## Optimum Scaling

Each amplifier or integrator used in the analog mechanization of the equations has a gain associated with each input. The coefficients of a problem uniquely determine the total loop gain through any $n^{th}$-order loop of the system where $1 \leq n \leq$ NEQ. While amplitude scaling cannot alter the total gain through any given $n^{th}$-order loop, it can insure against any excessively high or low gains by even distribution. If all $n^{th}$-order loops were independent of each other, calculation of the total loop gains of order $n$ and the $n^{th}$ roots of these total gains would suffice to describe optimum distribution of gains through each loop. Since the $n^{th}$-order loops are not independent, an optimization technique far too complex to perform manually is employed.

## Program Notation

The notation used for the vector whose $n$ components are the maximum values of the system is AMA(I), where I = 1, 2, 3,· · ·, NEQ. The elements of NTER(I) where I = 1, 2,· · ·, NEQ are the number of terms of each equation.

The gain at each amplifier is represented by the matrix PG(I, J); that is, the potentiometer-amplifier gain of each term in every equation. Each element of the PG matrix is calculated by the following equation:

$$\mathrm{PG(I,J)} = \frac{[\mathrm{A(I,J)}]\left\{\mathrm{AMA}[\mathrm{M(I,J)}]\right\}}{\mathrm{AMA(I)}} . \tag{1-a}$$

If the M(I, J) of any term indicates that two variables $V_1$ and $V_2$ are to be multiplied, then the equation is interpreted as

$$\mathrm{PG(I,J)} = \frac{[\mathrm{A(I,J)}]\left[\mathrm{AMA}(V_1)\cdot\mathrm{AMA}(V_2)\right]}{\mathrm{AMA(I)}} . \tag{1-b}$$

Optimum scaling, defined previously as minimizing the deviation from unity of the gain at each amplifier, is accomplished by minimizing the merit value, $\phi$, described by

$$\phi = \sum_{\mathrm{I}=1}^{\mathrm{NEQ}} \sum_{\mathrm{J}=1}^{\mathrm{NTER(I)}} [\mathrm{PG(I,J)} - 1]^2 , \tag{2}$$

where (a) PG(I, J) is replaced by 1.0/PG(I, J) if PG(I, J) < 1.0 and (b) PG(I, J) is replaced by 1.0 if M(I, J) = I. Part (a) above assures that low gains will not be ignored in the minimization routine; for example, a PG gain of 1/4 contributes the same value to $\phi$ as a PG gain of 4. Part (b) says that pot gains required in first-order feedback loops are not to be included in the value of $\phi$. Since

$$\mathrm{PG(I,J)} = \frac{\mathrm{A(I,J)\,AMA}[\mathrm{M(I,J)}]}{\mathrm{AMA(I)}} = \frac{\mathrm{A(I,J)\,AMA(I)}}{\mathrm{AMA(I)}} = \mathrm{A(I,J)}$$

when M(I, J) = I, the pot gain is constant, not dependent on a maximum-value choice, and therefore not included in the calculation of $\phi$.

## Search Methods

The principal search routine used in finding the minimum merit value is the gradient or steepest-descent method. The gradient is obtained from the derivative of the function $\phi$ at some point in the space represented by the vector $\mathrm{AMA}_0\mathrm{(I)}$, I = 1, NEQ. Since $\phi$ is a multi-variable function, (a function of NEQ variables), partial differentiation is used. The gradient line or direction of steepest descent is represented parametrically by

$$\Delta\,\mathrm{AMA}_0\mathrm{(I)} = \frac{\partial\phi}{\partial\mathrm{AMA}_0\mathrm{(I)}}\,\lambda \qquad (\mathrm{I} = 1, 2, \cdots, \mathrm{NEQ}) , \tag{3}$$

where $\lambda$ is an arbitrary negative parameter.

The selection of an optimum $\lambda$, which determines the distance to move on the steepest-descent line, results in movement to a new point in space, $\mathrm{AMA}_1(\mathrm{I})$, calculated from

$$\mathrm{AMA}_1(\mathrm{I}) = \mathrm{AMA}_0(\mathrm{I}) + \Delta\,\mathrm{AMA}_0(\mathrm{I}) \qquad (\mathrm{I} = 1, 2, \cdots, \mathrm{NEQ}) . \tag{4}$$

If the system of equations does not have product terms, the new merit value $\phi_1$ associated with the new point in space, $\mathrm{AMA}_1(\mathrm{I})$, is calculated from Equations 1-a, 2, 3, and 4 to obtain

$$\phi_1 = \sum_{\mathrm{I}=1}^{\mathrm{NEQ}} \sum_{\mathrm{J}=1}^{\mathrm{NTER(I)}} \left\{ \frac{\mathrm{A(I, J)} \left\{ \mathrm{AMA}_0 \left[\mathrm{M(I, J)}\right] + \Delta\,\mathrm{AMA}_0 \left[\mathrm{M(I, J)}\right] \right\}}{\mathrm{AMA}_0(\mathrm{I}) + \Delta\,\mathrm{AMA}_0(\mathrm{I})} - 1 \right\}^2 \tag{5-a}$$

or simplified

$$\phi_1 = \sum_{\mathrm{I}=1}^{\mathrm{NEQ}} \sum_{\mathrm{J}=1}^{\mathrm{NTER(I)}} \left\{ \frac{\mathrm{A(I, J)} \left\{ \mathrm{AMA}_1 \left[\mathrm{M(I, J)}\right] \right\}}{\mathrm{AMA}_1(\mathrm{I})} - 1 \right\}^2 . \tag{5-b}$$

For a nonlinear system with multipliers, Equations 1-b, 2, 3, and 4 result in

$$\phi_1 = \sum_{\mathrm{I}=1}^{\mathrm{NEQ}} \sum_{\mathrm{J}=1}^{\mathrm{NTER}} \left\{ \frac{\mathrm{A(I, J)} \left[\mathrm{AMA}_0(\mathrm{V}_1) + \Delta\,\mathrm{AMA}_0(\mathrm{V}_1)\right] \left[\mathrm{AMA}_0(\mathrm{V}_2) + \Delta\,\mathrm{AMA}_0(\mathrm{V}_2)\right]}{\mathrm{AMA}_0(\mathrm{I}) + \Delta\,\mathrm{AMA}_0(\mathrm{I})} - 1 \right\}^2 \tag{6-a}$$

or simplified

$$\phi_1 = \sum_{\mathrm{I}=1}^{\mathrm{NEQ}} \sum_{\mathrm{J}=1}^{\mathrm{NTER}} \left\{ \frac{\mathrm{A(I, J)} \left[\mathrm{AMA}_1(\mathrm{V}_1) \cdot \mathrm{AMA}_1(\mathrm{V}_2)\right]}{\mathrm{AMA}_1(\mathrm{I})} - 1 \right\}^2 \tag{6-b}$$

where $\mathrm{V}_1$ and $\mathrm{V}_2$ are the variables to be multiplied.

Since $\phi_1$ in both Equations 5 and 6 is a function of the independent variable $\lambda$ in Equation 3 and has only one degree of freedom, a uni-dimensional search routine is used to find the value of $\lambda$ that minimizes $\phi_1$. This routine consists of selecting trial values of $\lambda$ within the interval being searched, evaluating $\phi_1$ for these trial values, and reducing the interval. These three procedures take place according to a specified algorithm.

Wilde (1964) discusses one-dimensional search procedures and includes a table showing reduction ratios for various sequential search plans. The reduction ratios show the Fibonacci search to be the most efficient, followed closely by the Golden Section routine. The Fibonacci and Golden Section procedures are very similar once the first experiment (trial value) has been determined. The basic difference is that in the Fibonacci search the number of experiments is selected a priori and enters into the calculation of the first experiment, while the starting point for the Golden Section is determined by a fixed ratio with the number of experiments dependent upon a defined error criterion. These techniques are effective in finding the minimum value of a function if it is not multimodal (more than one minimum) in the interval to be searched. If the function does have more than one minimum in the interval, it is possible that the search may not locate the minimum having the lowest value of the function.

The search procedure used in the program to find the optimum $\lambda$ is the Fibonacci technique using twenty experiments. The Fibonacci number associated with twenty experiments is 10,946; this says that after twenty experiments the original interval of uncertainty is reduced to less than 0.0001 times its original length.

The first step in the Fibonacci search is to establish a lower bound and an upper bound of the $\lambda$ interval. The lower bound, BD1, is set at 0 for all searches; the upper bound, BD2, is recalculated for each search. Since maximum values must be greater than zero and since $\lambda$ is always a negative parameter, a unique upper bound, BD2, is determined before each search by the following:

$$\text{If} \quad Z_I = \text{AMA}_0\,(I) \Big/ \frac{\partial \phi}{\partial\, \text{AMA}_0\,(I)} \qquad (I = 1, 2, \cdots, \text{NEQ}) \; ,$$

then

$$\text{BD2} = \text{MINIMUM}\left(Z_I > 0.0\right) \qquad (I = 1, 2, \cdots, \text{NEQ}) \; . \tag{7}$$

Also, $Z_I$ is negative only if the partial derivative, $\partial\phi/\partial\,\text{AMA}_0\,(I)$, is negative. Re-examination of Equations 3 and 4 shows that maximums with negative partials cannot become negative and therefore are eliminated in the calculation of BD2 in Equation 7.

The next step is to select the first experiment, EXP 1, in the established interval. Wilde (pp. 29-30) gives the following formula for obtaining $X_1$, the ratio of the interval [BD1, EXP 1] to the whole interval [BD1, BD2] :

$$X_1 = \frac{F_{n-1}}{F_n} + \frac{(-1)^n \epsilon}{F_n} \; , \tag{8}$$

where $n$ is the number of experiments, $F_n$ is the reduction ratio for $n$ experiments from the table mentioned previously, and $\epsilon$ is the minimum separation between any two experiments. Since the rate of change of $\phi$ as a function of $\lambda$ is very diverse, it would be difficult to choose a constant $\epsilon$ appropriate for all searches. Computer experimentation showed that the formula,

$$X_1 = \frac{F_n}{F_{n+1}} \; , \tag{9}$$

was very close to the precision of Equation 8 with $n = 20$, and satisfactorily handled the varied slope field of the $\phi$ function.

The length of the interval to be searched is

$$L = BD2 - BD1 ; \tag{10}$$

while the value for EXPERIMENT 1, the first trial value of $\lambda$, is

$$EXP\,1 = BD1 + L \cdot X_1 . \tag{11}$$

The search is continued by placing EXP 2 symmetric about the midpoint of the interval [BD1, BD2] with respect to EXP 1. Calculation of the $\phi$ values with $\lambda$ = EXP 1 and $\lambda$ = EXP 2 allows the experimenter to discard a portion of the interval, either [BD1, EXP 2] or [EXP 1, BD2] depending on which experiment has the minimum $\phi$ value. In Figure 1 with the function $\phi$ as shown, the segment [BD1, EXP 2] would be eliminated from the interval since $\phi(EXP\,2) > \phi(EXP\,1)$. Thus, the new BD1 is EXP 2 and EXP 3 is placed symmetric about the midpoint of the new interval [BD1, BD2] with respect to EXP 1. After $n$ experiments, the final selection of $\lambda$ is calculated from

$$\lambda = \frac{BD1 + BD2}{2} . \tag{12}$$

With the optimum $\lambda$ known, Equations 3 and 4 can be solved for the new point in the space, $AMA_1(I)$, where $I = 1, 2, \cdots, NEQ$. Following the same procedure, the vector $AMA_2(I)$ is found. This cycle continues until the new point is as close to the true optimum point as the precision of the program allows.

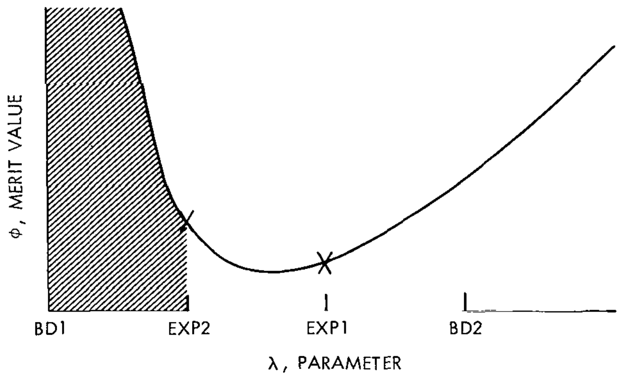


Figure 1—Graph of merit value versus arbitrary negative parameter.

In general, slope (sensitivity) of the merit value as a function of an AMA component varies inversely with the magnitude of the component. Extremely small AMA's have very steep slopes. Calculation of $\lambda$ for any steepest-descent movement in which a given equation or equations have small maximum values and large positive partial derivatives shows that $\lambda$ must be extremely small in order to satisfy Equation 7, which assures that no maximums will become negative or zero. This obviously becomes a limiting factor on the rate of convergence to the optimum point. Even if all small maximums had negative partials, the size of $\lambda$ would still be limited owing to the steep slopes involved.

Since most linear and any nonlinear system of equations with a moderate range of coefficients necessitates presence of small maximums, a solution to the limitations of steepest descent is offered in the form of a three-step, modified steepest-descent movement:

*Step 1:* a. AMA's $\leq 1.0$ are held constant

b. AMA's $> 1.0$ move in modified steepest descent direction until $\phi$ is minimized.

*Step 2:* a. AMA's $\leq 0.1$ and AMA's $> 1.0$ are held constant

b. $0.1 <$ AMA's $\leq 1.0$ move as in step 1(b).

*Step 3:* a. AMA's $> 0.1$ are held constant

b. AMA's $\leq 0.1$ move as in step 1(b).

If at any time in the search, the 20-step Fibonacci routine does not provide enough precision for the merit value to be steadily decreased, the program automatically increases the number of steps of the $\lambda$ search. This feature is desirable, particularly when the neighborhood of the optimum point is reached.

Although the modified steepest descent is fast and sufficiently handles most problems, an auxiliary search was programmed which can be activated and deactivated by sense switch 5, turned on and off manually on the computer console. The auxiliary search is a combination sequential, linear-interpolation routine, in which the partial derivative or slope of each maximum less than one is individually minimized with respect to all other maximums held constant. While in most cases this is too time-consuming to be a part of the normal search, it is useful near the beginning of a search or at any time that the modified gradient method is moving slowly, perhaps indicating approach to a local minimum. The auxiliary search provides a boost so that the modified gradient can again be effective.

## Special Systems

A LCC (linear constant coefficient) system of equations allows a great deal of flexibility in the optimization of scaling. A suggested approach for automatic scaling is to use only the required input as discussed under "Structure—Required Input," letting the program supply the initial AMA vector.

Since all LCC systems have an infinite number of AMA vectors associated with the optimum-merit test, the resulting AMA vector may then be multiplied by any scalar and still maintain the same minimized merit value and corresponding potentiometer settings. This property of a linear system enables the engineer to keep peak voltages near the ±100-volt level simply by adjusting the scalar quantity.

A linear system with variable coefficients, while not permitted the freedom of a LCC system, does have an infinite number of AMA vectors that have the same minimum $\phi$ value. However, since these maximum vectors are not proportional, as they are in the LCC case, the engineer must rely on the search routine to find different vectors with the same minimum $\phi$.

Nonlinear systems have only one maximum-value (AMA) vector having the minimum merit value. This vector will be located by the search if no constraints or constant maximums are specified as input. If constraints are placed on some maximums or if any maximums are determined and used as input, scaling will be optimum but subject to the input constraints. If limits and nonlinearities other than multipliers are in the system, it is suggested that the engineer examine the nonlinearities and place constraints or constant maximums where they are necessary or beneficial.

### Output

The output provides documentation of (1) all data used as input to the program, (2) the results of the optimum scaling—maximum values, levels, potentiometer readings and gains (printed sequentially by equation), and (3) the minimized $\phi$ value.

Obviously, the levels chosen by the program, although optimum for the analog simulation, are not convenient rounded numbers. Since in most cases the engineer prefers scaling that will allow him to quickly change from scaled to problem units, perhaps even mentally, the output includes an additional feature in which all levels are rounded to one significant figure, barring any which had been specified as constant in the input. Revised documentation of (2) and (3) listed above is printed. The increased $\phi$ value provides simple evaluation of the rounding effect.

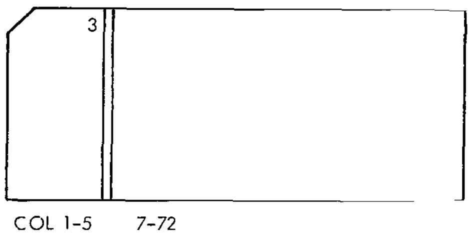
A data card is punched for each equation, giving the equation number and the rounded maximum value in the format accepted as input to the program. This deck is invaluable to the programmer for any future rescaling that may be required by additional constraints or changes or simply for new documentation.

All input prepared for the Automatic Scaling Program (NEQ, A MATRIX, M MATRIX) and the AMA deck (punched output of the program discussed in the preceding paragraph) also is in the proper format for the FAST Program, a digital program which provides static and dynamic solutions as a checkout for the analog.

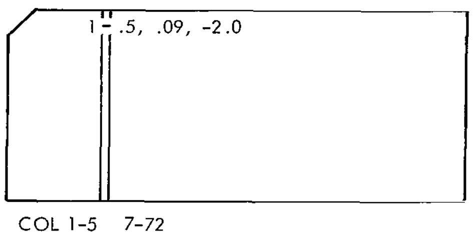
## OPERATING INSTRUCTIONS

### Required Input

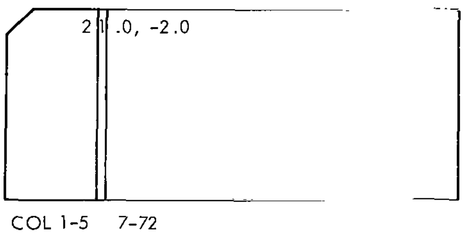
Assuming the three-equation problem with the equations, coefficients, and variables as discussed under "Structure—Required Input," the following cards are required. Data card 1 contains NEQ entered as an integer, right-justified in columns 1-5. Data cards 2-4 contain the A(I, J) matrix, one card for each equation. In card 2, for instance, columns 1-5 contain the equation number, right-justified, integer form. The coefficient A(1, 1) is entered in floating point starting in



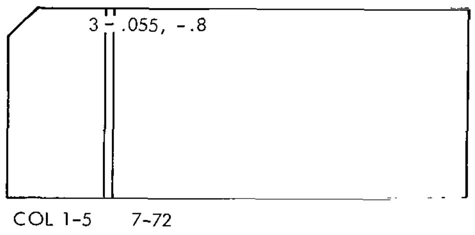
Card 1



Card 2



Card 3

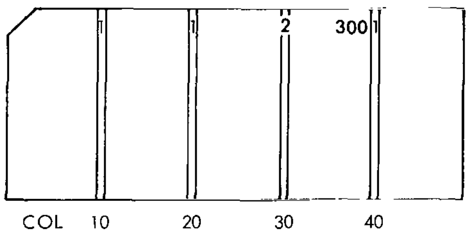


Card 4

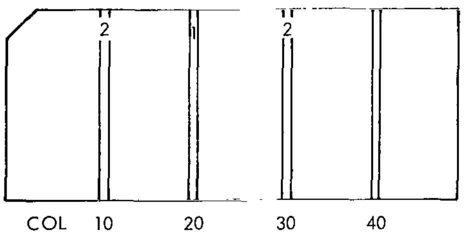
column 6 and terminated by a comma. The column to the immediate right of the comma contains the sign or first digit of A(1, 2). A(1, 2) is similarly terminated by a comma. This process continues until all coefficients of the equation have been entered. Each coefficient including signs and decimal point must not exceed 15 columns.

Data cards 5-7 contain the M(I, J) matrix, one equation per card. The equation number in interger form is right-justified in columns 1-10 and is followed by the M(I, 1), M(I, 2), M(I, 3) terms, all integer, right-justified in columns 11-20, 21-30, 31-40.

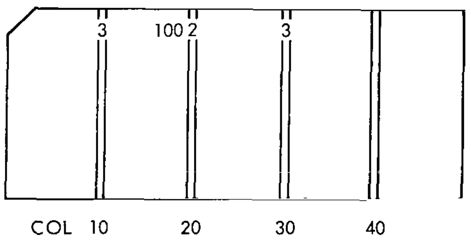
Card 5 indicates the format for multipliers. The integer 3001 right-justified in the 10 columns allowed for the M(1, 3) shows that the variable of term (1, 3) is the product, $X_3 \cdot X_1$.



Card 5



Card 6



Card 7

## Optional Input

The optional input is entered as two different data sets described by:

*Set 1:* Contains a starting set of maximums and indicates which, if any, of these values are to be held constant.

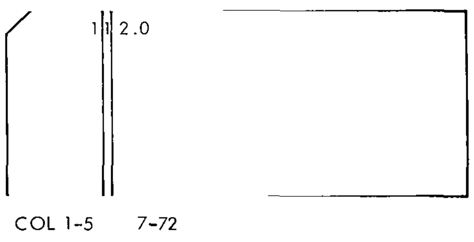
*Set 2:* Specifies all minimum and maximum constraints placed on any AMA's.

A blank card must be at the end of each data set to indicate termination of the set unless values are given for every equation. Even if a data set is omitted entirely, the blank card is still needed.

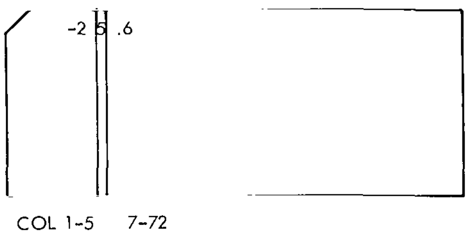
Assuming the same problem used as an example in the required input section, suppose that the programmer wants the maximums of Equations 1 and 2 to have the values 12.0 and 5.6 at the beginning of the search. Suppose also that the 5.6 value is to be held constant throughout the search. These values are found on cards 8 and 9. Columns 1-5 contain the equation number in integer form and right-justified. The maximum appears in floating point starting in column 6, not to exceed column 21. Any maximums to be held constant must have a minus sign preceding the equation number.

Since Equation 3 is not supplied with a starting or constant maximum, the program sets its maximum equal to 10.0 before the search is begun. Card 10 must be a blank card to show the end of optional input, Set 1.

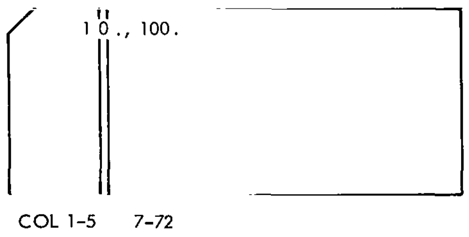
Suppose that the AMA (1) should not exceed 100.0 and the AMA (3) should be greater than 5.0 but less than 50.0. These constraints are found on cards 11 and 12. The equation number is entered as an integer, right-justified in columns 1-5. The constraints are entered as floating-point numbers; first the minimum which starts in column 6, next a comma, and finally the maximum. If there is no maximum constraint, the comma is not necessary.



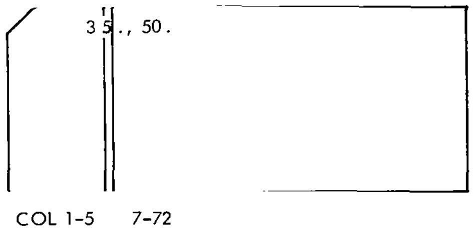
Card 8



Card 9



Card 11



Card 12

Card 13, a blank card indicating the end of optional input Set 2, is the final card of the input.

## Sense–Switch Operation

Table 1 shows the programming controlled by sense switches 1-5. The switches marked "*" are particularly valuable. Under normal conditions all switches are turned off.

Table 1

Programming Controlled by Sense Switches 1-5.

| S.S. Number | "ON" Operation |
|---|---|
| 1 | Prints documentation after each search. |
| 2 | Prints Fibonacci steps and Auxiliary Search. |
| 3 | Resets program for more than one set of data. |
| *4 | Designed to allow the programmer to choose his own precision; "ON" state terminates the program, provides complete documentation. |
| *5 | Activates auxiliary search. |

## FUTURE DEVELOPMENTS

Future developments will focus on additional flexibility and ease in programming nonlinearities.

Division of variables will be allowed with standard input form similar to the present form of multiplier input.

Beside permitting constant maximums and constraints as now programmed, future development will include the option of stating constant ratios to be maintained between two or more maximum values. This will be helpful in eliminating potentiometers and handling built-in gains in switching networks and other equipment.

If mathematical time-scaling is beneficial, this must, at present, be precalculated and reflected in the coefficient matrix. In the future the engineer will be given the option to experiment with different time-scaling factors by simply providing the factors as input. The program will evaluate the merit of time-scaling by comparing all minimized $\phi$ values.

Goddard Space Flight Center
National Aeronautics and Space Administration
Greenbelt, Maryland, August 1969
604-31-03-12-51

## REFERENCES

Wilde, D. J., "Optimum Seeking Methods," Englewood Cliffs, N. J.: Prentice Hall, Inc., 1964, pp. 10-52.

## Appendix A

## Automatic Scaling Example

The sample problem displayed in this section was run using the Fortran program listed in Appendix B developed specifically for the SDS 9300 digital computer. The sample program shows scaling of the set of three differential equations previously discussed in both sections dealing with required input.

```
AUTOMATIC SCALING EXAMPLE   (NO OPTIONAL INPUT)
 NEQ = 3

 A MATRIX

    1        -.50000000        .09000000    -2.00000000         .00000000
    2        1.00000000      -2.00000000      .00000000         .00000000
    3        -.05500000       -.80000000      .00000000         .00000000

 M MATRIX

          1         1         2       3001
          2         1         2
          3      1002         3

 MAXIMUMS


 CONSTRAINTS
 AUTOMATIC SCALING EXAMPLE                                                              DATE 18 APR 1969     PAGE     0002
       EQ            AMA(I)             DELTA(I)            LEVEL(I)              TERM1               TERM2               TERM3               TERM4
      +++    ++++++++++++++     ++++++++++++++      ++++++++++++++      ++++++++++++++      ++++++++++++++      ++++++++++++++      ++++++++++++++


        1        10.000000                 .00           10.000000             .500000             .090000          20.000000


        2        10.000000                 .00           10.000000            1.000000            2.000000


        3        10.000000                 .00           10.000000             .550000             .800000



      MERIT TEST   =             463.903989389

      NO.    1      MERIT =            47.6914760942
      NO.    4      MERIT =            31.8816575337
      NO.    7      MERIT =            20.1429973378
      NO.   10      MERIT =            17.1616107626
      NO.   13      MERIT =            15.1862255049
      NO.   16      MERIT =            14.6474871758
      NO.   19      MERIT =            14.3417070574
      NO.   22      MERIT =            14.2702215434
      NO.   25      MERIT =            14.2284847689
      NO.   28      MERIT =            14.2226775511
      NO.   31      MERIT =            14.2155651171
      NO.   34      MERIT =            14.1997667058
      NO.   37      MERIT =            14.0170847617
      NO.   38      MERIT =            14.0161128830
      NO.   40      MERIT =            12.8622146225
      NO.   41      MERIT =            12.8622130664
      NO.   43      MERIT =            12.7508967381
      NO.   44      MERIT =            12.7477292105
      NO.   46      MERIT =            12.7426480996
      NO.   47      MERIT =            12.7420924535
      NO.   49      MERIT =            12.7411870731
      NO.   50      MERIT =            12.7410791704
      NO.   52      MERIT =            12.7408452642
      NO.   53      MERIT =            12.7408163005
      NO.   55      MERIT =            12.7407442527
      NO.   56      MERIT =            12.7407348893
      NO.   58      MERIT =            12.7407102898
      NO.   59      MERIT =            12.7407072995
      NO.   61      MERIT =            12.7406987791
      NO.   62      MERIT =            12.7406975928
      NO.   64      MERIT =            12.7406941927
      NO.   65      MERIT =            12.7406937588
      NO.   67      MERIT =            12.7406924722
      NO.   68      MERIT =            12.7406923282
```

```
AUTØMATIC SCALING EXAMPLE
         NØ.  70        MERIT =            12.7406919070
         NØ.  71        MERIT =            12.7406918531
         NØ.  73        MERIT =            12.7406916970
         NØ.  74        MERIT =            12.7406916693
         NØ.  76        MERIT =            12.7406915765
         NØ.  77        MERIT =            12.7406915749
         NØ.  79        MERIT =            12.7406915453
         NØ.  80        MERIT =            12.7406915699
         NØ.  80        MERIT =            12.7406915446
         NØ.  82        MERIT =            12.7406916012
         NØ.  82        MERIT =            12.7406915399
         NØ.  83        MERIT =            12.7406915650
         NØ.  83        MERIT =            12.7406915393
         NØ.  85        MERIT =            12.7406916117
         NØ.  85        MERIT =            12.7406915374
         NØ.  86        MERIT =            12.7406915658
         NØ.  86        MERIT =            12.7406915372
         NØ.  88        MERIT =            12.7406916197
         NØ.  88        MERIT =            12.7406915364
         NØ.  89        MERIT =            12.7406915669
         NØ.  89        MERIT =            12.7406915363
         NØ.  89        MERIT =            12.7406915363
         NØ.  89        MERIT =            12.7406915363
         NØ.  91        MERIT =            12.7406916267
         NØ.  91        MERIT =            12.7406915362
         NØ.  92        MERIT =            12.7406915647
         NØ.  92        MERIT =            12.7406915359
         NØ.  94        MERIT =            12.7406916283
         NØ.  94        MERIT =            12.7406915358
         NØ.  95        MERIT =            12.7406915684
         NØ.  95        MERIT =            12.7406915358
         NØ.  95        MERIT =            12.7406915358
         NØ.  95        MERIT =            12.7406915358
         NØ.  97        MERIT =            12.7406916315
         NØ.  97        MERIT =            12.7406915358
         NØ.  97        MERIT =            12.7406915358
         NØ.  97        MERIT =            12.7406915358
         NØ.  98        MERIT =            12.7406915684
         NØ.  98        MERIT =            12.7406915358
         NØ.  98        MERIT =            12.7406915358
         NØ.  98        MERIT =            12.7406915358
         NØ. 100        MERIT =            12.7406916315
         NØ. 100        MERIT =            12.7406915358
         NØ. 100        MERIT =            12.7406915358
         NØ. 100        MERIT =            12.7406915358
```

```
AUTOMATIC SCALING EXAMPLE                                                   DATE 18 APR 1969    PAGE     0004
   EQ            AMA(I)            DELTA(I)            LEVEL(I)            TERM1              TERM2              TERM3              TERM4
  +++  ++++++++++++++  ++++++++++++++  ++++++++++++++  ++++++++++++++  ++++++++++++++  ++++++++++++++  +++++++++++++


    1          3.035474             1.81           32.943779            .500000            .322968           1.907138


    2         10.892903             1.53            9.180290            .278665           2.000000


    3           .953569             -.00          104.869206           1.907138            .800000



   MERIT TEST  =                12.740691536

 MINIMUM MERIT VALUE




AUTOMATIC SCALING EXAMPLE                                                   DATE 18 APR 1969    PAGE     0005
   EQ            AMA(I)            DELTA(I)            LEVEL(I)            TERM1              TERM2              TERM3              TERM4
  +++  ++++++++++++++  ++++++++++++++  ++++++++++++++  ++++++++++++++  ++++++++++++++  ++++++++++++++  +++++++++++++


    1          3.333333             1.81           30.000000            .500000            .300000           2.000000


    2         11.111111             1.53            9.000000            .300000           2.000000


    3          1.000000             -.00          100.000000           2.037037            .800000



   MERIT TEST  =                12.964334705
 ALL LEVELS ARE ROUNDED

 *STOP* 00001000
```

Appendix B

# Fortran Program

```
      INTEGER NEQ,NTER(150),M(150,4),C(150)
      DOUBLE PRECISION  A(150,4),AMA(0:150),CT(150,2),DELTA(0:150),
     *                  PG(150,4),SAMA(150),SDELTA(150),XAMA(150),XDELTA
     *                  (150),XLEV(150),YAMA(150),YDELTA(150)
      DOUBLE PRECISION  ABS,BD1,BD2,BD3,BMAX,CDELTA,D,D1,D2,DIFF,EXP1,
     *                  EXP2,FACT,GK,MERIT,SGK,SIN,SIS,SMERIT,SUBT,SUM,
     *                  SUMP,V,X,X1PHI,X2PHI,Y,Z,ZZ
C
***********************************************************************
C
C     AUTOMATIC ANALOG SCALING
C
***********************************************************************
C     SEC 1 - READ IN NEQ, COEFFICIENTS, AND M MATRIX
***********************************************************************
      ABS(X)=DABS(X)
      READ(105, 10) NEQ
   10 FORMAT(I5,4(F16.8))
      OUTPUT(108) NEQ,'','','A MATRIX',''
      DO 30  I=1,NEQ
      READ(105, 10) C(I), (A(I,J),J=1,4)
      WRITE(108, 10) C(I),(A(I,J),J=1,4)
      IF(C(I).NE.I) GO TO 1370
      NTER(I)=1
      DO 20 J=1,4
      IF(A(I,J).EQ.0.) GO TO 30
      NTER(I)=NTER(I)+1
   20 CONTINUE
   30 CONTINUE
      OUTPUT(108)'','','M MAIRIX',''
      DO 50 I=1,NEQ
      READ(105, 40) C(I),(M(I,J),J=1,NTER(I)-1)
      WRITE(108, 40) C(I),(M(I,J),J=1,NTER(I)-1)
   40 FORMAT(5I10)
      IF(C(I).NE.I) GO TO 1370
   50 CONTINUE
***********************************************************************
C     SEC 2 - RESET CONDITIONS FOR NEW AMA SET AND CONSTRAINTS
***********************************************************************
   60 SENSE LIGHT 0,3,15
      NUM=MUM=NAC=NOD=LAST=MAST=0
      BD3=1.
      AMA(0)=1.0
      SMERIT=.9*(10.**20)
      CDELTA=0.0
   70 DO 80  I=1,NEQ
      SDELTA(I)=0.0
      AMA(I)=12345.
      C(I)=+1
      DO 80 J=1,2
   80 CT(I,J)=0.
***********************************************************************
C     SEC 3 - READ IN INITIAL MAXIMUM VALUES
```

```
***********************************************************************
      ØUTPUT(108)'','','MAXIMUMS',''
      DØ 100  I=1,NEQ
      READ(105, 90) N,V
      IF(N.EQ.0) GØ TØ 110
      WRITE(108, 90) N,V
   90 FØRMAT(I5,F16.8)
      L=IABS(N)
      AMA(L)=V
      C(L)=N
  100 CØNTINUE
  110 DØ 120 I=1,NEQ
      IF(AMA(I).NE.12345.)GØ TØ 120
      AMA(I)=10.0
  120 CØNTINUE
***********************************************************************
C     SEC 4 - READ IN CØNSTRAINTS
***********************************************************************
      ØUTPUT(108)'','','CØNSTRAINTS',''
      DØ 140 I=1,NEQ
      READ(105, 130) N,(CT(N,J),J=1,2)
      IF(N.EQ.0) GØ TØ 150
      WRITE(108, 130) N,(CT(N,J),J=1,2)
  130 FØRMAT(I5,2(F16.8))
  140 CØNTINUE
***********************************************************************
C     SEC 5 - CALCULATIØN ØF MERIT VALUE
***********************************************************************
  150 DØ 170  I=1,NEQ
      DØ 170  J=1,NTER(I)-1
      LL=M(I,J)/1000
      NN=MØD(M(I,J),1000)
  160 PG(I,J)=(AMA(LL)*AMA(NN)*A(I,J))/AMA(I)
      PG(I,J)=ABS(PG(I,J))
  170 CØNTINUE
      MERIT=0.
      DØ 200  I=1,NEQ
      DØ 200 J=1,NTER(I)-1
      IF(M(I,J) .EQ. I) GØ TØ 200
      IF(PG(I,J).GE.1.0) GØ TØ 180
      Y= (1.0/PG(I,J))-1.0
      GØ TØ 190
  180 Y=PG(I,J)-1.0
  190 MERIT=MERIT + (Y**2)
  200 CØNTINUE
***********************************************************************
C     SEC 6 - PRINT AMAS, LEVELS, PØT GAINS, MERIT VALUE
***********************************************************************
  210 IF(SENSE LIGHT 3) 260, 220
  220 CØNTINUE
      IF(SENSE SWITCH 2) 230, 250
  230 WRITE(108, 240) LY,MERIT
  240 FØRMAT(5X,$LY=$,I3,$,MERIT=$,F25.9)
  250 IF(SENSE LIGHT 10) 1080, 1100
```

```
 260 IF(SENSE LIGHT 15) 270, 370
 270 WRITE(108, 280)
 280 FORMAT(1H1,5X,$EQ$,11X,$AMA(I)$,10X,$DELTA(I)$,10X,$LEVEL(I)$,13X,
    *     $TERM1$,13X,$TERM2$,13X,$TERM3$,12X,$TERM4$)
     WRITE(108, 290)
 290 FORMAT(5X,3($+$),2X,(7(15($+$),3X)))
     DO 340  I=1,NEQ
     XLEV(I)=100./AMA(I)
     OUTPUT(108) ' '
     IF((MAST.EQ.1).OR.(LAST.EQ.1)) GO TO 300
     GO TO 340
 300 IF((MAST.EQ.1).AND.(C(I).LT.0)) GO TO 330
 310 WRITE(106, 320) I,AMA(I)
 320 FORMAT(I5,F16.8)
     GO TO 340
 330 IT=-I
     WRITE(106, 320) IT,AMA(I)
 340 WRITE(108, 350) I,AMA(I),SDELTA(I),XLEV(I),(PG(I,J),J=1,NTER(I)-1)
 350 FORMAT(2/,5X,I3,2X,F15.6,F18.2,3X,(5(F15.6,3X)))
     WRITE(108, 360) MERIT
 360 FORMAT(5/,5X,$MERIT TEST  =$,F24.9)
     IF(LAST.EQ.1) GO TO 1450
     IF(MAST.EQ.1) GO TO 1400
     OUTPUT(108) ' ',' '
     SENSE LIGHT 2
 370 IF(SENSE LIGHT 2) 400, 380
 380 WRITE(108, 390) NOD,MERIT
 390 FORMAT(5X,$NO.$,I4,5X,$MERIT = $,F21.10)
************************************************************************
C     SEC 7 - COMPARE NEW MERIT WITH MINIMUM MERIT, STORE MINIMUM MERIT
************************************************************************
 400 IF(SENSE LIGHT 14) 1390, 410
 410 IF((SMERIT-MERIT) .GT. (10.**(-10))) GO TO 460
     IF(NAC.NE.2) GO TO 430
     DO 420 I=1,NEQ
 420 AMA(I)=YAMA(I)
     NAC=0
     GO TO 570
 430 NUM=NUM+1
     IF(NUM.LT.4) GO TO 1050
     DO 440 I=1,NEQ
 440 AMA(I)=YAMA(I)
     MUM=MUM+1
     NUM=0
     GO TO 510
 450 SENSE LIGHT 3,14,15,16
     GO TO 1250
 460 SMERIT=MERIT
     DO 470 I=1,NEQ
     YAMA(I)=AMA(I)
     YDELTA(I)=SDELTA(I)
 470 CONTINUE
     NUM=MUM=0
     IF((MOD(NOD,3).EQ.2).AND.(NAC.EQ.2))NOD=NOD+1
```

```
************************************************************************
C     SEC 8 - SENSE SWITCH TO TERMINATE PROGRAM
************************************************************************
      IF(SENSE SWITCH 4) 480, 490
  480 SENSE LIGHT 3,15
      MAST=1
      GO TO 150
************************************************************************
C     SEC 9 - SENSE SWITCH FOR INTERPOLATION OF AMAS LESS THAN ONE
************************************************************************
  490 IF(SENSE SWITCH 5) 500, 510
  500 NAC=1
      GO TO 520
  510 NAC=0
      IF(MUM.GT.3) GO TO 450
************************************************************************
C     SEC 10 - CALCULATION OF DELTAS
************************************************************************
  520 NOD=NOD+1
  530 IF(MOD(NOD,3).EQ.1) GO TO 570
      IF(MOD(NOD,3).EQ.0) GO TO 550
      DO 540 I=1,NEQ
      IF((AMA(I).GT.0.1).AND.(AMA(I).LE.1.0)) GO TO 570
  540 CONTINUE
      GO TO 520
  550 DO 560 I=1,NEQ
      IF(AMA(I).LE.0.1) GO TO 570
  560 CONTINUE
      GO TO 520
  570 DO 580 I=1,NEQ
      DELTA(I)=0.0
      DO 580 J=1,NTER(I)-1
      NN=MOD(M(I,J),1000)
      LL=M(I,J)/1000
      IF(PG(I,J).LT.1.0) X=(2.*((AMA(I)/(AMA(LL)*AMA(NN)*A(I,J)))-1.))*
     *     (1./(AMA(LL)*AMA(NN)*A(I,J)))
      IF(PG(I,J).GE.1.0) X=(2.*(((AMA(LL)*AMA(NN)*A(I,J))/AMA(I))-1.0))*
     *     (-1.*((AMA(LL)*AMA(NN)*A(I,J))/(AMA(I)**2)))
  580 DELTA(I)=DELTA(I)+X
      DO 590  I=1,NEQ
      DO 590 J=1,NTER(I)-1
      LL=M(I,J)/1000
      NN=MOD(M(I,J),1000)
      IF(PG(I,J).LT.1.0) Z=(2.*((AMA(I)/(AMA(LL)*AMA(NN)*A(I,J)))-1.))
     *     *((-1. *AMA(I))/(A(I,J)*AMA(NN)*(AMA(LL)**2)))
      IF(PG(I,J).GE.1.0) Z=(2.*(((AMA(LL)*AMA(NN)*A(I,J))/AMA(I))-1.0))
     *     *((A(I,J)*AMA(NN))/AMA(I))
  590 DELTA(LL)=DELTA(LL)+Z
      DO 600 I=1,NEQ
      DO 600 J=1,NTER(I)-1
      LL=M(I,J)/1000
      NN=MOD(M(I,J),1000)
      IF(PG(I,J).LT.1.0) ZZ=(2.*((AMA(I)/(AMA(LL)*AMA(NN)*A(I,J)))-1.))
     *     *((-1. *AMA(I))/(A(I,J)*AMA(LL)*(AMA(NN)**2)))
```

```
      IF(PG(I,J).GE.1.0) ZZ=(2.*((( AMA(LL)*AMA(NN)*A(I,J))/AMA(I))-1.))
     *      *((A(I,J)*AMA(LL))/AMA(I))
  600 DELTA(NN)=DELTA(NN)+ZZ
*************************************************************************
C     SEC 11 - INTERPOLATION ROUTINE ACTIVATED BY SENSE SWITCH 5
*************************************************************************
      IF(NAC.EQ.0) GO TO 970
      IF(MOD(NOD,3).EQ.1) GO TO 970
  610 IF(SENSE LIGHT 11) 730, 620
  620 IF(SENSE LIGHT 19) 800, 630
  630 NIT=0
      REPEAT 900,WHILE((BMAX(AMA,DELTA,NEQ,C).GT.(.01)).AND.(NIT.LE.20))
      DO 890  L=1,NEQ
      MAD=MAR=0
      SIS=SIN=.01
      MM=MMM=0
      FACT=2.0
      SENSE LIGHT 0
      IF(C(L).LT.0) GO TO 890
      IF(AMA(L).GT.1.) GO TO 890
      IF(ABS(DELTA(L)).LE.(.01)) GO TO 890
      XAMA(L)=AMA(L)
      XDELTA(L)=DELTA(L)
  640 AMA(L)=XAMA(L)
      IF(SENSE LIGHT 12) 680, 650
  650 AMA(L)=AMA(L)+(AMA(L)*SIS)
      IF(SENSE SWITCH 2) 660, 670
  660 OUTPUT(108) 'AMA + .01',AMA(L)
  670 GO TO 720
  680 AMA(L)=AMA(L)-(AMA(L)*SIN)
  690 IF(SENSE SWITCH 2) 700, 710
  700 OUTPUT(108) 'AMA - .01',AMA(L)
  710 SENSE LIGHT 17
  720 SENSE LIGHT 11
      GO TO 570
  730 IF(SENSE SWITCH 2) 740, 750
  740 OUTPUT(108)XDELTA(L),DELTA(L)
  750 IF((XDELTA(L)*DELTA(L)).GT.0.) GO TO 820
      SUM=ABS(XDELTA(L))+ABS(DELTA(L))
      DIFF=ABS(XAMA(L)-AMA(L))
      SUMP= ABS(XDELTA(L))/SUM
      IF(SENSE SWITCH 2) 760, 770
  760 OUTPUT(108) XAMA(L),SUMP,DIFF
  770 IF(XAMA(L).GT.AMA(L)) AMA(L)=XAMA(L)-(SUMP*DIFF)
      IF(XAMA(L).LT.AMA(L))AMA(L)=XAMA(L)+(SUMP*DIFF)
      SENSE LIGHT 23
      IF(SENSE SWITCH 2) 780, 790
  780 OUTPUT(108) AMA(L),L
  790 SENSE LIGHT 19
      GO TO 570
  800 MMM=MMM+1
      IF(SENSE LIGHT 23) 810, 890
  810 IF((ABS(DELTA(L)).LE.(.1)).OR.(MMM.NE.1)) GO TO 890
      IF(AMA(L).GT.1.0) GO TO 890
      SENSE LIGHT 0
      MM=MAD=MAR=0
      SIS=SIN=.01
      FACT=2.0
      XAMA(L)=AMA(L)
      XDELTA(L)=DELTA(L)
      GO TO 650
  820 IF(AMA(L).GE.CT(L,1)) GO TO 830
      AMA(L)=CT(L,1)
      C(L)=-L
      CT(L,1)=0.
      OUTPUT(108) 'MINIMUM CONSTRAINT IN ITERATION'
      GO TO 790
  830 IF(AMA(L).GE.(.00001)) GO TO 840
      AMA(L)=1.E-5
      GO TO 790
  840 IF(AMA(L).GE.1.0) GO TO 790
      IF(SENSE LIGHT 17) 860, 850
  850 IF(ABS(DELTA(L)).GE.ABS(XDELTA(L))) GO TO 870
      MAR=MAR+1
      IF((MAR.EQ.1).OR.(MAR.EQ.2)) SIS=SIS*10.
      IF(MAR.GE.3) SIS=SIS*2.
      GO TO 640
  860 IF(ABS(DELTA(L)).LT.ABS(XDELTA(L))) GO TO 880
  870 MM=MM+1
      IF(MM.NE.2) GO TO 880
      AMA(L)=XAMA(L)
      GO TO 790
  880 SENSE LIGHT 12
      MAD=MAD+1
      IF(MAD.EQ.2) SIN=SIN*10.
      IF(MAD.LE.2) GO TO 640
      IF(MAD.GE.5) FACT=1.0+((FACT-1.0)/2.0)
      SIN=SIN*FACT
      GO TO 640
  890 CONTINUE
  900 NIT=NIT+1
      IF(SENSE SWITCH 2) 910, 920
  910 OUTPUT(108) '91 CONTINUE'
  920 SENSE LIGHT 3
      NAC=2
  930 IF(SENSE SWITCH 1) 940, 950
  940 SENSE LIGHT 15
  950 DO 960 I=1,NEQ
  960 SDELTA(I)=DELTA(I)
      GO TO 150
*************************************************************************
C     SEC 12 - STORE AMAS AND DELTAS
*************************************************************************
  970 DO 980 I=1,NEQ
      SDELTA(I)=DELTA(I)
      SAMA(I)=AMA(I)
  980 CONTINUE
*************************************************************************
```

```
C     SEC 13 - ESTABLISH UPPER BOUND OF LAMBDA
****************************************************************************
  990 MAX=0
      IF(MOD(NOD,3).EQ.2) BD3=1.E-5
      IF(MOD(NOD,3).EQ.0) BD3=1.E-8
      DO 1020 LX=1,NEQ
      IF(SDELTA(LX).LE.0.) GO TO 1020
      IF(C(LX).LT.0) GO TO 1020
      IF((MOD(NOD,3).EQ.1).AND.(SAMA(LX).LE.1.0)) GO TO 1020
      IF((MOD(NOD,3).EQ.2).AND.((SAMA(LX).GT.1.0).OR.(SAMA(LX).LE.0.1)))
     *  , GO TO 1020
      IF((MOD(NOD,3).EQ.0).AND.(SAMA(LX).GT..1))GO TO 1020
 1000 MAX=MAX + 1
      GK=ABS(AMA(LX)/SDELTA(LX))
      IF(MAX.NE.1) GO TO 1010
      SGK=GK
      GO TO 1020
 1010 IF(SGK.GT.GK) SGK=GK
 1020 CONTINUE
      IF(MAX.EQ.0) BD2=BD3
      IF(MAX.NE.0) BD2=SGK
      BD1=0.
      BD3=BD2
      IF(SENSE SWITCH 2) 1030, 1050
 1030 WRITE(108, 1040)BD1,BD2
 1040 FORMAT(5X,$BD1=$,F20.9,$, BD2=$,F20.9)
****************************************************************************
C     SEC 14 - FIBONACCI SEARCH FOR OPTIMUM LAMBDA
****************************************************************************
 1050 DO 1190  LY=1,19
      IF(LY.NE.1) GO TO 1060
      DIFF=BD2-BD1
      D2=10946./17711.
      D1=D2*DIFF
      EXP1=BD1+D1
      EXP2=BD2-D1
      SENSE LIGHT8
      SENSE LIGHT 9
 1060 IF(SENSE LIGHT 8) 1070, 1090
 1070 CDELTA=EXP1
      SENSE LIGHT 10
      GO TO 1250
 1080 X1PHI=MERIT
      IF(SENSE LIGHT 9) 1090, 1110
 1090 CDELTA=EXP2
      GO TO 1250
 1100 X2PHI=MERIT
 1110 IF(X2PHI.GT.X1PHI) GO TO 1150
      BD2=EXP1
      EXP1=EXP2
      SUBT=BD2-EXP1
      EXP2=BD1+SUBT
      X1PHI=X2PHI
      IF(SENSE SWITCH 2) 1120, 1140
 1120 WRITE(108, 1130) EXP1,EXP2,BD1,BD2
 1130 FORMAT(5X,$CASE1$,F18.9,F18.9,$,BD1=$,F15.9,$,BD2=$,F15.9)
 1140 IF(LY.EQ.19) GO TO 1200
      GO TO 1190
 1150 BD1=EXP2
      EXP2=EXP1
      SUBT=EXP2-BD1
      EXP1=BD2-SUBT
      X2PHI=X1PHI
      SENSE LIGHT 8
      IF(SENSE SWITCH 2) 1160, 1180
 1160 WRITE(108, 1170) EXP1,EXP2,BD1,BD2
 1170 FORMAT(5X,$CASE2$,F18.9,F18.9,$,BD1=$,F15.9,$,BD2=$,F15.9)
 1180 IF(LY.EQ.19) GO TO 1200
 1190 CONTINUE
 1200 CDELTA=(BD2+BD1)/2.0
      SENSE LIGHT 3
      IF(SENSE SWITCH 1) 1210, 1220
 1210 SENSE LIGHT 15
 1220 CONTINUE
      IF(SENSE SWITCH 2) 1230, 1250
 1230 WRITE(108, 1240) CDELTA
 1240 FORMAT(5X,$FINAL CDELTA=$,F20.9)
***************************************************************************
C     SEC 15 - CALCULATION OF AMAS WITH OPTIMUM LAMBDA
***************************************************************************
 1250 DO 1340 I=1,NEQ
      IF(SENSE LIGHT 16) 1320, 1260
 1260 IF(C(I).LT.0) GO TO 1340
      IF((MOD(NOD,3).EQ.1).AND.(SAMA( I).LE.1.0)) GO TO 1340
      IF((MOD(NOD,3).EQ.2).AND.((SAMA( I).GT.1.0).OR.(SAMA( I).LE.0.1)))
     *    GO TO 1340
      IF((MOD(NOD,3).EQ.0).AND.(SAMA( I).GT..1))GO TO 1340
      DELTA(I)=SDELTA(I)*(CDELTA)
 1270 AMA(I)=SAMA(I) -DELTA(I)
      IF((CT(I,1).EQ.0.).AND.(CT(I,2).EQ.0.)) GO TO 1340
      IF((CT(I,1).NE.0.).AND.(CT(I,2).NE.0.)) GO TO 1290
      IF(CT(I,1).EQ.0.) GO TO 1280
      IF(AMA(I).GE.CT(I,1))GO TO 1340
      GO TO 1300
 1280 IF(AMA(I).LE.CT(I,2)) GO TO 1340
      GO TO 1310
 1290 IF((AMA(I).GE.CT(I,1)).AND.(AMA(I).LE.CT(I,2))) GO TO 1340
      IF(AMA(I).GT.CT(I,2)) GO TO 1310
 1300 C(I)=-I
      AMA(I)=CT(I,1)
      CT(I,1)=0.
      SENSE LIGHT 4
      GO TO 1340
 1310 C(I)=-I
      AMA(I)=CT(I,2)
      CT(I,2)=0.
      SENSE LIGHT 4
      GO TO 1340
```