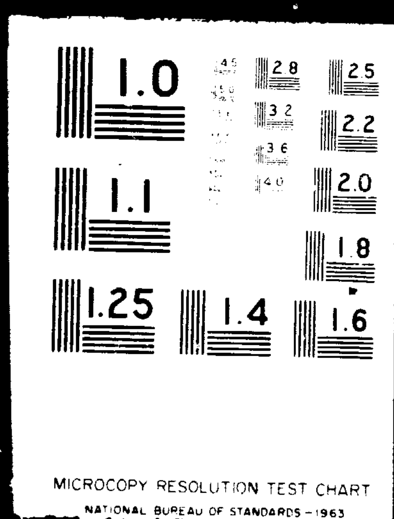
1 OF 1

N73 26619 UNCLAS



NASA TECHNICAL MEMORANDUM

NASA TM X- 62,282

NASA TM X- 62,282

(NASA-TM-X-62282) CONMIN: A FORTRAN PROGRAM FOR CONSTRAINED FUNCTION MINIMIZATION: USER'S MANUAL (NASA)
~~61~~ 63 p HC $5.25
CSCL 12A

N73-26619

Unclas
G3/19 08415

CONMIN – A FORTRAN PROGRAM FOR CONSTRAINED FUNCTION MINIMIZATION USER'S MANUAL

Garret N. Vanderplaats

Ames Research Center
and
U.S. Army Air Mobility R&D Laboratory
Moffett Field, Calif. 94035

August 1973

CONTENTS

LIST OF FIGURES

SUMMARY

CONMIN is a FORTRAN program, in subroutine form, for the solution of linear or non-linear constrained optimization problems. The basic optimization algorithm is the Method of Feasible Directions. The user must provide a main calling program and an external routine to evaluate the objective and constraint functions and to provide gradient information. If analytic gradients of the objective or constraint functions are not available, this information is calculated by finite difference. While the program is intended primarily for efficient solution of constrained problems, unconstrained function minimization problems may also be solved, and the conjugate direction method of Fletcher and Reeves is used for this purpose. This manual discribes the use of CONMIN and defines all necessary parameters. Sufficient information is provided so that the program can be used without special knowledge of optimization techniques. Sample problems are included to help the user become familiar with CONMIN and to make the program operational.

CONMIN - A FORTRAN PROGRAM FOR CONSTRAINED FUNCTION MINIMIZATION

SECTION 1

INTRODUCTION

In many mathematical problems, it is necessary to determine the minimum or maximum of a function of several variables, limited by various linear and nonlinear inequality constraints. It is seldom possible, in practical applications, to solve these problems directly, and iterative methods are used to obtain the numerical solution. Machine-calculation of this solution is, of course, desirable and the CONMIN program has been developed to solve a wide variety of such problems.

CONMIN is a FORTRAN program, in subroutine form, for the minimization of a multi-variable function subject to a set of inequality constraints. The general minimization problem is; Find values for the set of variables, X(I), to

Minimize OBJ

Subject to;

G(J).LE.0 J=1,NCON

VLB(I).LE.X(I).LE.VUB(I) I=1,NDV NSIDE.GT.0

where OBJ is a general function (objective function) of the variables, X(I), referred to hereafter as decision variables. OBJ need not be a simple analytic function, and may be any function which can be numerically evaluated.

G(J) is the value of the Jth inequality constraint, which is also a function of the X(I). NCON is the number of constraints, G(J). NCON may be zero. VLB(I) and VUB(I) are lower and upper bounds respectively on variable X(I), and are referred to as side constraints. NSIDE=0 indicates that no lower or upper bounds are prescribed. If NCON=0 and NSIDE=0, the objective function is said to be unconstrained. NDV is the total number of decision variables, X(I).

Constraint G(J) is defined as active if CT.LE.G(J).LE.ABS(CT), and violated if G(J).GT.ABS(CT), where constraint thickness, CT, is a small specified negative number. The numerical significance of CT may be understood by refering to Fig. 1, which shows a single constraint in a two variable design space. Constraint G(J) is mathematically equal to zero along a single line in the design space. However, on a digital computer, the exact value of G(J)=0 can seldom be obtained. Therefore, a constraint thickness of 2*ABS(CT) is chosen to define the region over which G(J) is assumed to be zero for purposes of optimization. Because all G(J) must be negative, CT is taken as a negative number for consistancy so that any G(J).GT.CT is defined as active (or violated) if G(J).GT.ABS(CT). While it may seem logical to choose a very small value (in magnitude) for CT (say -1.0E-6), the nature of the optimization algorithm used by CONMIN is such that more numerical stability can be achieved by taking CT=-0.1 or even -0.2. CT is used for numerical stability only, and when the optimization is complete, one or more constraints, G(J), will usually be very near zero, as seen in the examples in SECTION VIII.

It is desirable that G(J) be normalized so that

-1.LE.G(J).LE.1   J=1,NCON

In this way the constraint thickness, CT, has the same numerical significance for all G(J). It is not necessary that all G(J) be precisely in this form, and such normalization may not be possible. However, it is important that all G(J) at least be of the same order of magnitude. For example, assume that some G(J) =X(1)**2-X(2). If X(1) and X(2) are expected to be of order 100 for the particular problem under consideration, G(J) may be scaled by dividing by 10,000 to provide a value for G(J) of order one.

The basic analytic technique used by CONMIN is to minimize OBJ until one or more constraints, G(J), become active. The minimization process then continues by following the constraint boundaries in a direction such that the value of OBJ continues to decrease. When a point is reached such that no further decrease in OBJ can be obtained, the process is terminated. The value of the constraint thickness parameter, CT, and the normalization of the constraints, G(J), have considerable effect on the numerical stability and rate of convergence of the optimization process.

An example of a constrained nonlinear problem is the minimization of the four variable Rosen-Suzuki function (ref. 1);

```
MINIMIZE  OBJ = X(1)**2 - 5*X(1) + X(2)**2 - 5*X(2) +
                2*X(3)**2 - 21*X(3) + X(4)**2 + 7*X(4) + 50
```

Subject to;

```
G(1) = X(1)**2 + X(1) + X(2)**2 - X(2) +
       X(3)**2 + X(3) + X(4)**2 - X(4) - 8 .LE. 0
```

```
G(2)  =  X(1)**2 - X(1) + 2*X(2)**2 +X(3)**2 +
         2*X(4)**2 - X(4) - 10                    .LE. 0
G(3)  =  2*X(1)**2 + 2*X(1) + X(2)**2 - X(2) +
         X(3)**2 - X(4) - 5                       .LE. 0
```

This problem has four decision variables and three constraints, (NDV=4, NCON=3). No lower or upper bounds VLB(I) or VUB(I) are prescribed so control parameter NSIDE is specified as NSIDE=0 to indicate this. It is necessary to provide a set of initial values for X(I), and from this the constrained optimum is obtained by CONMIN and its associated routines. This problem will be solved using CONMIN in SECTION VIII.

The minimization algorithm is based on Zoutendijk's method of feasible directions (ref. 2). The algorithm has been modified to improve efficiency and numerical stability and to solve optimization problems in which one or more constraints, G(J), are initially violated (ref. 3). While the program is intended primarily for the efficient solution of constrained functions, unconstrained functions may also be minimized (NCON=0, and NSIDE=0), and the conjugate direction method of Fletcher and Reeves (ref. 4) is used for this purpose. If a function is to be maximized, this may be achieved by minimizing the negative of the function.

For constrained minimization problems, the initial design need not be feasible (one or more G(J) may be greater than ABS(CT)), and a feasible solution (if one exists) is obtained with a minimal increase in the value of the objective function.

The user must supply a main program to call subroutine CONMIN along with an external subroutine to evaluate the objective function, constraint functions and the analytic gradient of the objective and currently active or violated constraint functions. At any given time in the minimization process, gradient information is required only for constraints which are active or violated (G(J).GE.CT). Gradients are calculated by finite difference if this information is not directly obtainable, and a subroutine is included with CONMIN for this purpose.

The basic program organization is described here, and sufficient information is provided so that the program may be used without special knowledge of optimization techniques. The various control parameters are described and the required dimensions of all arrays are given so that the user can limit storage requirements to that necessary to solve his particular problems. Sample problems are included to aid the user in making the program operational and to gain familiarity with its use.

A summary of all parameters used by CONMIN and its associated routines is given in APPENDIX A for convenient reference.

APPENDIX B contains a brief discription of the subroutines associated with CONMIN.

SECTION II

MAKING CONMIN OPERATIONAL

CONMIN utilizes iterative techniques to numerically obtain the minimum of a general function, subject to a prescribed set of linear and/or non-linear constraints. Because of the nature of these techniques, the efficiency of the optimization process (the number of times the functions must be evaluated) can often be improved by the proper choice of control parameters to deal effectively with a given problem. For this reason, it is particularly desirable that the new user solve several simple two to five variable problems, experimenting with different control parameters. The following steps are suggested to help the user make CONMIN operational and to gain the familiarity necessary for its efficient application to a particular problem.

1. Obtain the deck, including example problems.
2. Read SECTION VIII (EXAMPLES) of this manual.
3. Solve the example problems using CONMIN, and verify the results by comparison with the output listed in this manual. Note that the precise numerical values may differ slightly on different computers.
4. Read this entire manual carefully.
5. Devise several two to five variable unconstrained and constrained minimization problems and solve them using CONMIN. If the precise optimum can be determined analytically, compare this with the optimums obtained using CONMIN.
6. Experiment by starting from several different initial points (different initial X vectors). This is good practice in all optimization problems, since it improves the chances that the

absolute minimum is obtained (instead of a relative minimum).

7. Experiment with various analytic gradient options by solving the same problems with and without calculating precise analytic gradients (see examples 1 and 2 of Section VIII).
8. Experiment with various convergence criteria, DELFUN, DABFUN and ITRM.
9. Experiment with various constraint thickness parameters, CT, CTMIN, CTL and CTLMIN to understand the effect of these parameters on constrained minimization problems.
10. Experiment with various values of the push-off factor, THETA, on several constrained minimization problems. Small values of THETA may be used if constraints, G(J), are nearly linear functions of the decision variables, X(I), and larger values should be used if one or more G(J) are highly non-linear.
11. Experiment with scaling options by using no scaling, automatic scaling, or user provided scaling options.
12. Experiment with various conjugate direction restart parameters, ICNDIR, using examples of unconstrained minimization. Note that if ICNDIR=1, the steepest descent method will be used throughout the optimization process.
13. Re-read this entire manual.

The default options on the various control parameters have been chosen as reasonable values for most optimization problems. The steps listed above are intended to provide the user with the experience necessary to change these parameters as required to efficiently solve new optimization problems of special interest.

SECTION III

PROGRAM ORGANIZATION

The overall program organization is shown schematically in Fig. 2. The variables are initialized in the main program. The vector X is iteratively changed in subroutine CONMIN and its associated routines, and external routine SUB1 calculates the required function values and gradient information.

Subroutine CONMIN is called by the main program by the call statement;

CALL CONMIN(SUB1,OBJ)

where SUB1 is the name of the user supplied external subroutine and OBJ is the optimum value of the objective function upon return from CONMIN. The variables, X(I), contained in vector X and constraint values, G(J), contained in vector G will correspond to this optimum upon return. If additional information calculated in external routine SUB1 is required, the routine should be called again upon return from CONMIN to insure that this information corresponds to the final values of X(I).

Subroutine SUB1 is called by CONMIN and its associated routines by the call statement;

CALL SUB1(INFO,OBJ)

where INFO is a control parameter defining which information must be supplied, and OBJ is the value of the objective function to be calculated corresponding to the current decision variables contained in X. INFO will have a value of from one to four to identify what information must be supplied by SUB1.

INFO=1 Calculate objective function value, OBJ, for current variables X.

INFO=2 Calculate objective function value, OBJ, and constraint values, G(J), J=1,NCON for current variables, X.

INFO=3 Calculate analytic gradient of objective function corresponding to current variables, X. The objective function and constraint values already correspond to the current values of X and need not be recalculated. However, other infomation obtained in SUB1 when calculating OBJ and G(J) may not correspond to X and must be calculated again here if it is used in gradient computations. If finite difference control parameter, NFDG, is set to NFDG=1 in the main program this value of INFO will never be considered.

INFO=4 For current variables, X, determine which constraints are active and which are violated (G(J).GE.CT) and how many such constraints there are (NAC = Number of active and violated constraints). Calculate the analytic gradients of the objective function and all active or violated constraints. Values of the objective function, OBJ, and constraints, G(J), already correspond to the current variables, X, and need not be recalculated. As in the case of INFO=3, all other information used in gradient computations must be calculated for the current variables, X. If finite difference control parameter NFDG, defined in the main program, is not zero, this value of INFO will never be considered.

Note that INFO=3 and INFO=4 are considered only if gradient information is calculated in the user supplied subroutine, SUB1.

With the exception of the external subroutine name, SUB1, the objective function value, OBJ, and the control parameter, INFO, all information used in the optimization process is transferred by means of the following labeled common blocks.

```
      COMMON/CNMN1/IPRINT,NDV,ITMAX,NCON,NSIDE,ICNDIR,NSCAL,NFDG,
     1 FDCH,FDCHM,CT,CTMIN,CTL,CTLMIN,THETA,PHI,NAC,NACMX1,DELFUN,
     2 DABFUN,LINOBJ,ITRM,ITER,NCAL(4)
      COMMON/CNMN2/X(N1),DF(N1),G(N2),ISC(N8),IC(N4),A(N4,N3)
      COMMON/CNMN3/S(N3),G1(N7),G2(N2),C(N9),MS1(N6),B(N4,N4)
      COMMON/CNMN4/VLB(N1),VUB(N1),SCAL(N5)
```

The parameters and arrays used in the minimization process will now be defined, followed by the required array dimensions, N1 through N9. If a default value is listed, this value will be used if a zero value is transferred from the main program.

SECTION IV

PARAMETERS DEFINED IN MAIN PROGRAM

IPRINT Print control. All printing is done on file number 6.

0; Print nothing.

1; Print initial and final function information.

2; 1st debug level. Print all of above plus control parameters. Print function value and X-vector at each iteration.

3; 2nd. debug level. Print all of above plus all constraint values, numbers of active or violated constraints, direction vectors, move parameters and miscellaneous information. The constraint parameter, BETA, printed under this option approaches zero as the optimum objective is achieved.

4; Complete debug. Print all of above plus gradients of objective function, active or violated constraint functions and miscellaneous information.

NDV Number of decision variables, X(I), contained in vector X.

ITMAX Default value = 10. Maximum number of iterations in the minimization process. If NFDG.EQ.0 each iteration requires one set of gradient computations (INFO = 3 or 4) and approximately three function evaluations (INFO = 1 or 2). If NFDG.GT.0 each iteration requires approximately NDV+3 function evaluations (INFO = 1 or 2).

NCON Number of constraint functions, G(J). NCON may be zero.

NSIDE Side constraint parameter. NSIDE=0 signifies that the variables X(I) do not have lower or upper bounds. NSIDE.GT.0

signifies that all variables X(I) have lower and upper bounds defined by VLB(I) and VUB(I) respectively. If one or more variables are not bounded while others are, the values of the lower and upper bounds on the unbounded variables must be taken as very large negative and positive values respectively (ie. VLB(I)=-1.0E+10, VUB(I)=1.0E+10).

ICNDIR Default value = NDV+1. Conjugate direction restart parameter. If the function is currently unconstrained, (all G(J).LT.CT or NCON=NSIDE=0), Fletcher-Reeves conjugate direction method will be restarted with a steepest descent direction every ICNDIR iterations. If ICNDIR=1 only steepest descent will be used.

NSCAL Scaling control parameter. The decision variables will be scaled linearly.

NSCAL.LT.0; Scale variables X(I) by dividing by SCAL(I), where vector SCAL is defined by the user.

NSCAL.EQ.0; Do not scale the variables.

NSCAL.GT.0; Scale the variables every NSCAL iterations. Variables are normalized so that scaled X(I)=X(I)/ABS(X(I)). When using this option, it is desirable that NSCAL=ICNDIR if ICNDIR is input as nonzero, and NSCAL=NDV+1 in ICNDIR is input as zero.

NFDG Gradient calculation control parameter.

NFDG=0; All gradient information is provided by external routine SUB1. This information may be calculated analytically, or by finite difference, at the user's discretion.

NFDG=1; All gradient information will be calculated by finite difference in CONMIN. SUB1 provides only function values, OBJ and G(J), J=1,NCON.

NFDG=2; Gradient of objective function is provided by external routine SUB1, and gradients of active and violated constraints are calculated by finite difference in CONMIN. This option is desirable if the gradient of the objective function is easily obtained in closed form, but gradients of constraint functions, G(J), are unobtainable. This option may improve efficiency if several variables are limited by lower or upper bounds.

FDCH Default value = 0.01. Not used in NFDG=0. Relative change in decision variable X(I) in calculating finite difference gradients. For example, FDCH=0.01 corresponds to a finite difference step of one percent of the value of the decision variable.

FDCHM Default value = 0.01. Not used if NFDG = 0. Minimum absolute step in finite difference gradient calculations. FDCHM applies to the unscaled variable values.

CT Default value = -0.1. Not used if NCON=NSIDE=0. Constraint thickness parameter. If CT.LE.G(J).LE.ABS(CT), G(J) is defined as active. If G(J).GT.ABS(CT), G(J) is said to be violated. If G(J).LT.CT, G(J) is not active. CT is sequentially reduced in magnitude during the optimization process. If ABS(CT) is very small, one or more constraints may be active on one iteration and inactive on the next, only to become active again on a subsequent iteration. This is often referred to as 'zigzagging' between constraints. A wide initial value of the constraint thickness is desirable for highly nonlinear problems so that when a constraint becomes active it tends to remain active, thus reducing the zigzagging problem. The default value is usually adequate.

CTMIN Default value = 0.004. Not used if NCON=NSIDE=0. Minimum absolute value of CT considered in the optimization process. CTMIN may be considered as 'numerical zero' since it may not be meaningful to compare numbers smaller than CTMIN. The value of CTMIN is chosen to indicate that satisfaction of a constraint within this tolerance is acceptable. The default value is usually adequate.

CTL Default value = -0.01. Not used if NCON=NSIDE=0. Constraint thickness parameter for linear and side constraints. CTL is smaller in magnitude than CT because the zig-zagging problem is avoided with linear and side constraints. The default value is usually adequate.

CTLMIN Default value = 0.001. Not used if NCON=NSIDE=0. Minimum absolute value of CTL considered in the optimization process. The default value is usually adequate.

THETA Default value = 1.0. Not used if NCON=NSIDE=0. Mean value of the push-off factor in the method of feasible directions. A larger value of THETA is desirable if the constraints, G(J), are known to be highly non-linear, and a smaller value may be used if all G(J) are known to be nearly linear. The actual value of the push-off factor used in the program is a quadratic function of each G(J), varying from 0.0 for G(J)=CT to 4.0*THETA for G(J)=ABS(CT). A value of THETA=0.0 is used in the program for constraints which are identified by the user to be strictly linear. THETA is called a 'push-off' factor because it pushes the design away from the active constraints into the feasible region. The default value is usually adequate.

PHI Default value = 5.0. Not used if NCON=NSIDE=0. Participation coefficient, used if a design is infeasible (one or more G(J).GT.ABS(CT)). PHI is a measure of how hard the design will be 'pushed' towards the feasible region and is, in effect, a penalty parameter. If in a given problem, a feasible solution cannot be obtained with the default value, PHI should be increased, and the problem run again. If a feasible solution cannot be obtained with PHI=100, it is probable that no feasible solution exists. The default value is usually adequate.

NACMX1 Not used if NSIDE=NCON=0. 1 plus user's best estimate of the maximum number of constraints (including side constraints, VLB(I) and VUB(I)) which will be active at any given time in the minimization process. NACMX1 = number of rows in array A. If NAC+1 ever exceeds this value, the minimization process will be terminated, an error message will be printed, and control will return to the main program. NACMX1 will never exceed NDV+1 if all constraints G(J) and bounds VLB(I) and VUB(I) are independent. A reasonable value for NACMX1 (and the corresponding dimension of array A) is MIN(40,NDV+1), where the minimum of 40 will only apply for large problems and is arbitrary, based on the observation that even for very large problems (over a hundred X(I) and several thousand G(J)), it is uncommon for many constraints to be active at any time in the minimization process (the optimum solution is seldom 'fully constrained' for very large nonlinear problems).

DELFUN Default value = 0.001. Minimum relative change in the objective function to indicate convergence. If in ITRM consecutive iterations, ABS(1.0-OBJ(J-1)/OBJ(J)).LT.DELFUN and the current design is feasible (all G(J).LE.ABS(CT)), the minimization process is terminated. If the current design is infeasible (some G(J).GT.ABS(CT)), five iterations are required to terminate and this situation indicates that a feasible design may not exist.

DABFUN Default value = 0.001 times the initial function value. Same as DELFUN except comparison is on absolute change in the objective function, ABS(OBJ(J)-OBJ(J-1)), instead of relative change.

LINOBJ Not used if NCON=NSIDE=0. Linear objective function identifier. If the objective, OBJ, is specifically known to be a strictly linear function of the decision variables, X(I), set LINOBJ=1. If OBJ is a general nonlinear function, set LINOBJ=0.

ITRM Default value = 3. Number of consecutive iterations to indicate convergence by relative or absolute changes, DELFUN or DABFUN.

X(N1) Vector of decision variables, X(I),I=1,NDV. The initial X-vector contains the user's best estimate of the set of optimum design variables.

VLB(N1) Used only if NSIDE.NE.0. VLB(I) is the lower allowable value (lower bound) of variable X(I). If one or more variables, X(I), do not have lower bounds, the corresponding VLB(I) must be initialized to a very large negative number (say -1.0E+10).

VUB(N1) Used only if NSIDE.NE.0. VUB(I) is the maximum allowable value (upper bound) of X(I). If one or more variables, X(I), do not have upper bounds, the corresponding VUB(I) must be initialized to a very large positive number (say 1.0E+10).

SCAL(N5) Not used if NSCAL=0. Vector of scaling parameters. If NSCAL.GT.0 vector SCAL need not be initialized since SCAL will be defined in CONMIN and its associated routines. If NSCAL.LT.0, vector SCAL is initialized in the main program, and the scaled variables X(I)=X(I)/SCAL(I). Efficiency of the optimization process can sometimes be improved if the variables are either normalized or are scaled in such a way that the partial derivative of the objective function, OBJ, with respect to variable X(I) is of the same order of magnitude for all X(I). SCAL(I) must be greater than zero because a negative value of SCAL(I) will result in a change of sign of X(I) and possibly yield erroneous optimization results. The decision of if, and how, the variables should be scaled is highly problem dependent, and some experimentation is desirable for any given class of problems.

ISC(N8) Not used if NCON=0. Linear constraint identification vector. If constraint G(J) is known to be a linear function of the decision variables, X(I), ISC(I) should be initialized to ISC(I)=1. If constraint G(J) is nonlinear ISC(I) is initialized to ISC(I)=0. Identification of linear constraints may improve efficiency of the optimization process and is therefore desirable, but is not essential. If G(J) is not specifically known to be linear, set ISC(I)=0.

SECTION V

PARAMETERS DEFINED IN EXTERNAL ROUTINE SUB1.

OBJ Value of objective function for the current decision variables, X(I),I=1,NDV contained in vector X. Calculate OBJ if INFO=1 or INFO=2.

G(N2) Not used if NCON=NSIDE=0. Vector containing all constraint functions, G(J), J=1,NCON for current decision variables, X. Calculate G(J), J=1,NCON if INFO=2.

DF(N1) Analytic gradient of the objective function for the current decision variables, X(I). DF(I) contains the partial derivative of OBJ with respect to X(I). Calculate DF(I),I=1,NDV if INFO=3 or INFO=4 and if NFDG=0 or NDFG=2.

NAC Number of active and violated constraints (G(J).GE.CT). Calculate NAC if INFO=4 and NFDG=0.

A(N4,N3) Not used if NCON=NSIDE=0. Gradients of active or violated constraints, for current decision variables, X(I). A(J,I) contains the gradient of the Jth active or violated constraint, G(J), with respect to the Ith decision variable, X(I) for J=1,NAC and I=1,NDV. Calculate if INFO=4 and NFDG=0.

IC(N4) Identifies which constraints are active or violated. IC(J) contains the number of the Jth active or violated constraint for J=1,NAC. For example, if G(10) is the first active or violated constraint (G(J).LT.CT, J=1,9), set IC(1)=10. Calculate if INFO=4 and NFDG=0.

If it is convenient to calculate more information than is required by the information control parameter, INFO, this may be done. INFO identifies the minimum amount of information which is necessary at a given time in the optimization process. It is never necessary to determine which bounds (side constraints) VLB(I) and VUB(I) are active because this information is determined by CONMIN.

The required organization of SUB1 is shown in Fig. 3. Note that if NCON=0, NFDG=1 or NFDG=2, much of Fig. 3 is inapplicable and can be omitted.

SECTION VI

PARAMETERS DEFINED IN CONMIN AND ASSOCIATED ROUTINES

ITER Iteration number. The optimization process is iterative so that the vector of decision variables at the Kth iteration is defined by X(K)=X(K-1)+ALPHA*S(K), where in this case K refers to the iteration number and the components X(I) are all changed simultaneously. ALPHA is defined as the move parameter and is printed if the print control IPRINT.GE.3. S is the move direction at iteration number J.

NCAL(4) Bookkeeping information. NCAL(1) gives the number of times external routine SUB1 was called with INFO=1. NCAL(2) gives the number of times INFO=2. NCAL(3) gives the number of times INFO=3 and NCAL(4) gives the number of times INFO=4.

S(N3) Move direction in the NDV-dimensional optimization space. S(I) gives the rate at which variable X(I) changes with respect to ALPHA.

G1(N7) Not used if NCON=NSIDE=NSCAL=0. Used for temporary storage of constraint values G(J), J=1,NCON and decision variables X(I), I=1,NDV.

G2(N2) Not used if NCON=NSIDE=0. Used for temporary storage of constraint values G(J), J=1,NCON.

B(N4,N4) Not used if NCON=NSIDE=0. Used in determining direction vector S for constrained minimization problems. Array B may be used for temporary storage in external routine SUB1.

C(N9) Not used in NCON=NSIDE=0. Used with array B in determining direction vector S for constrained minimization problems. Used for temporary storage of vector X if NSCAL.NE.0. routine SUB1.

MS1(N6) Not used if NCON=NSIDE=0. Used with array B in determining direction vector S for constrained minimization problems. Array MS1 may be used for temporary storage in external routine SUB1.

SECTION VII

REQUIRED DIMENSIONS OF ARRAYS

```
COMMON/CNMN2/X(N1),DF(N1),G(N2),ISC(N8),IC(N4),A(N4,N3)
COMMON/CNMN3/S(N3),G1(N7),G2(N2),C(N9),MS1(N6),B(N4,N4)
COMMON/CNMN4/VLB(N1),VUB(N1),SCAL(N5)
```

Dimensions N1 through N7 are minimum dimensions on the arrays and are defined by;

```
N1 = NDV
N2 = 1 if NCON=NSIDE=0
   = NCON if NCON.GT.0 and NSIDE=0
   = 2*NDV if NCON=0 and NSIDE.GT.0
   = NCON+2*NDV if NCON.GT.0 and NSIDE.GT.0
N3 = NDV+2
N4 = 1 if NCON=NSIDE=0
     NACMX1 if NCON.GT.0 or NSIDE.GT.0
N5 = 1 if NSCAL=0
   = N1 if NSCAL.NE.0
N6 = 1 if NCON=NSIDE=0
   = 2*N4 if NCON.GT.0 or NSIDE.GT.0
N7 = N2 if NSCAL = 0
   = MAX(N2,NDV) if NSCAL.NE.0
N8 = 1 if NCON=0
   = NCON if NCON.GT.0
N9 = N4 if NFDG.EQ.0
   = MAX(N4,NDV) if NFDG.GT.0
```

SECTION VIII

EXAMPLES

In this section several examples are presented, together with results, to provide a better understanding of the program organization. In each case the default values are used for control parameters unless otherwise noted. The array dimensions are defined in the common blocks as follows:

```
COMMON /CNMN2/ X(50),DF(50),G(1100),ISC(1000),IC(51),A(51,52)
COMMON /CNMN3/ S(51),G1(1100),G2(1100),C(51),MS1(102),B(51,51)
COMMON /CNMN4/ VLB(50),VUB(50),SCAL(50)
```

The examples were solved using a CDC 7600 computer. The numerical results obtained using other computers may differ slightly from those obtained here.

EXAMPLE 1 - CONSTRAINED ROSEN-SUZUKI FUNCTION. NO GRADIENT INFORMATION.

Consider the minimization problem discussed in SECTION I:

```
MINIMIZE OBJ = X(1)**2 - 5*X(1) + X(2)**2 - 5*X(2) +
               2*X(3)**2 - 21*X(3) + X(4)**2 + 7*X(4) + 50
```

Subject to:

```
        G(1) = X(1)**2 + X(1) + X(2)**2 - X(2) +
               X(3)**2 + X(3) + X(4)**2 - X(4) - 8 .LE.0
        G(2) = X(1)**2 - X(1) + 2*X(2)**2 + X(3)**2 +
               2*X(4)**2 - X(4) - 10                .LE.0
```

```
G(3) = 2*X(1)**2 + 2*X(1) + X(2)**2 - X(2) +
       X(3)**2 - X(4) - 5                        .LE.0
```

This problem is solved beginning with an initial X-vector of

X = (1.0, 1.0, 1.0, 1.0)

The optimum design is known to be

OBJ = 6.000

and the corresponding X-vector is

X = (0.0, 1.0, 2.0, -1.0)

The print control parameter of IPRINT=2 is used and all gradients are calculated by finite difference (NFDG = 1). The maximum number of iterations is taken as ITMAX=40 to insure normal termination. The variables are not scaled, so NSCAL=0. The objective function is nonlinear, so LINOBJ=0. The control parameters are defined as:

IPRINT=2, NDV=4, ITMAX=40, NCON=3, NFDG=1, NACMX1=51,

NSIDE=ICNDIR=NSCAL=LINOBJ=ITRM=0.

FDCH=FDCHM=CT=CTMIN=CTL=CTLMIN=THETA=PHI=DELFUN=DABFUN=0.

All constraints are nonlinear so the linear constraint identification vector contains all zeros:

ISC(J) = 0   J=1,NCON

The main program and the external routine, EXMP1, are listed in Figs. 4 and 5 respectively, with the optimization results in Fig. 6. An optimum design of OBJ=6.0183 is obtained with the corresponding decision variables:

X = (0.0125, 0.9969, 1.9943, -1.0006)

Note that the unconstrained minimum of this function may be found by setting NCON=0 in the main program. The unconstrained minimum of OBJ=-30.0 may be found in this way, and this is left as an exercise.

An additional problem of interest is to set NCON=2 and, having found the optimum subject to these first two constraints only, increase NCON to 3 and call CONMIN again, to obtain the final optimum design. This is easily done by initially setting NCON=2 in the main program, then immediately after returning from CONMIN, set NCON=3 and call CONMIN again. It is not necessary to reinitialize the control parameters. This exercise demonstrates the capability of CONMIN to deal with initially infeasible designs, and such an option may be desirable when minimizing functions for which one or more constraints are difficult or time-consuming to evaluate. In this way, the optimization problem may be first solved by ignoring constraints which are particularly complex. These constraints are then checked to determine if they are violated. If not, the optimization is complete. If one or more such constraints are violated, they are added to the set of constraints, G(J), and CONMIN is called again to obtain the final optimum design. This approach cannot always be expected to be most efficient, but does merit consideration, especially when only moderate constraint violations are expected.

EXAMPLE 2 - CONSTRAINED ROSEN-SUZUKI FUNCTION WITH ANALYTIC GRADIENTS

The function minimized in EXAMPLE 1 is now solved by computing all analytic gradients in closed form. All control parameters are the same as before except that now NFDG=0 instead of NFDG=1. The gradient of the objective function with respect to the decision variables is:

$$\mathrm{del(OBJ)} = \left\{\begin{array}{c} 2*X(1)-5 \\ 2*X(2)-5 \\ 4*X(3)-21 \\ 2*X(4)+7 \end{array}\right\}$$

The gradients of the constraint functions are:

$$\mathrm{del(G(1))} = \left\{\begin{array}{c} 2*X(1)+1 \\ 2*X(2)-1 \\ 2*X(3)+1 \\ 2*X(4)-1 \end{array}\right\} \qquad \mathrm{del(G(2))} = \left\{\begin{array}{c} 2*X(1)-1 \\ 4*X(2) \\ 2*X(3) \\ 4*X(4)-1 \end{array}\right\}$$

$$\mathrm{del(G(3))} = \left\{\begin{array}{c} 4*X(1)+2 \\ 2*X(2)-1 \\ 2*X(3) \\ -1 \end{array}\right\}$$

The control program is the same as before (Fig. 4) except now NFDG=0, and the external routine is named EXMP2. Also, the print control is now taken as IPRINT=1 to provide only summary information. External routine, EXMP2, is listed in Fig. 7 and the optimization results in Fig. 8, where an optimum design of

OBJ=6.0066 is obtained with

X = (-0.0072, 1.0022, 2.0053, -0.9907)

The additional exercises described in example 1 may also be solved here, just as before.

EXAMPLE 3 - 3-BAR TRUSS.

As a final example, consider the 3-bar truss shown in Fig. 9. The structure is subjected to two symmetric, but independent load conditions, P1 and P2, as shown. The truss is to be designed for minimum weight, subject to stress limitations only, so that;

-15 KSI .LE. SIGIJ .LE. 20 KSI   I=1,3   J=1,2

where SIGIJ is the stress in member I under load condition J. While this is a very simple structure, it is of particular historical significance in the field of automated structural design, having been first used by Schmit (ref. 5) to demonstrate that an optimally designed structure may not be fully stressed. That is, one or more members may not be stressed to their maximum design stress under any of the applied load conditions.

The design variables are chosen as the member cross-sectional areas, A1 and A2, where A3=A1 due to symmetry. Then the objective function is;

OBJ = 10*RHO*(2*SQRT(2)*A1+A2)

where RHO is the material density (RHO=0.1). The stress state is defined by;

SIG11 = SIG32 = 20.*(SQRT(2.)*A1+A2)/(2.*A1*A2+SQRT(2.)*A1*A1)

SIG21 = SIG22 = 20.*SQRT(2.)*A1/(2.*A1*A2+SQRT(2.)*A1*A1)

SIG31 = SIG13 = -20.*A2/(2.*A1*A2+SQRT(2.)*A1*A1)

Remembering that -15 KSI .LE. SIGIJ .LE. 20 KSI, there are six independent nonlinear constraints. The compressive stress constraint on member 1 under load condition 1 is given as;

-SIG11 - 15.0 .LE. 0

or in normalized form;

-SIG11/15.0 -1 .LE. 0

Similarly;

SIG11/20.0 -1 .LE. 0

-SIG21/15.0 -1 .LE. 0

SIG21/20.0 -1 .LE. 0

-SIG31/15.0 -1 .LE. 0

SIG31/20.0 -1 .LE. 0

Because negative member areas are not physically meaningful, lower bounds of zero must be imposed on the design variables. However, noting that the stress, SIGIJ, is undefined if A1 equals zero, lower bounds of 0.001 will be prescribed. The upper bounds are taken as 1.0E+10 to insure that these bounds will never be active.

The objective function is linear in A1 and A2 so the linear objective function identifier is taken as LINOBJ = 1.

The gradient of OBJ is easily calculated so this will be done analytically, while the gradients of the constraint functions are calculated by finite difference. Then NFDG = 2 and the gradient of OBJ is defined by:

$$\text{del(OBJ)} = \left\{ \begin{array}{c} 20.0*\text{SQRT}(2.0)*\text{RHO} \\ \\ 10.0*\text{RHO} \end{array} \right\}$$

The print control will be taken as IPRINT = 3 and the default values are used for all other control parameters. Then the control parameters are defined as;

IPRINT=3, NDV=3, NCON=6, NSIDE=1, NFDG=1, NACMX1=51, LINOBJ=1.

ITMAX=ICNDIR=NSCAL=ITRM=0.

FDCH=FDCHM=CT=CTMIN=CTL=CTLMIN=THETA=PHI=DELFUN=DABFUN=0.

All constraints are nonlinear so the linear constraint identification vector contains all zeros:

ISC(J) = 0   J = 1,NCON

The lower and upper bounds are defined as;

VLB(I) = 0.001   VUB(I) = 1.0E+10   I = 1,NDV

The optimum design is known to be

OBJ = 2.639

where

X = (0.789, 0.408)

The main program for this example is listed in Fig. 10 with the corresponding external routine, EXMP3, in Fig. 11. The optimization results are given in Fig. 12, where;

OBJ = 2.639

and

X = (0.789, 0.406)

Note that only constraint number 2 (the tensile stress constraint in member 1 under load condition 1) is active.

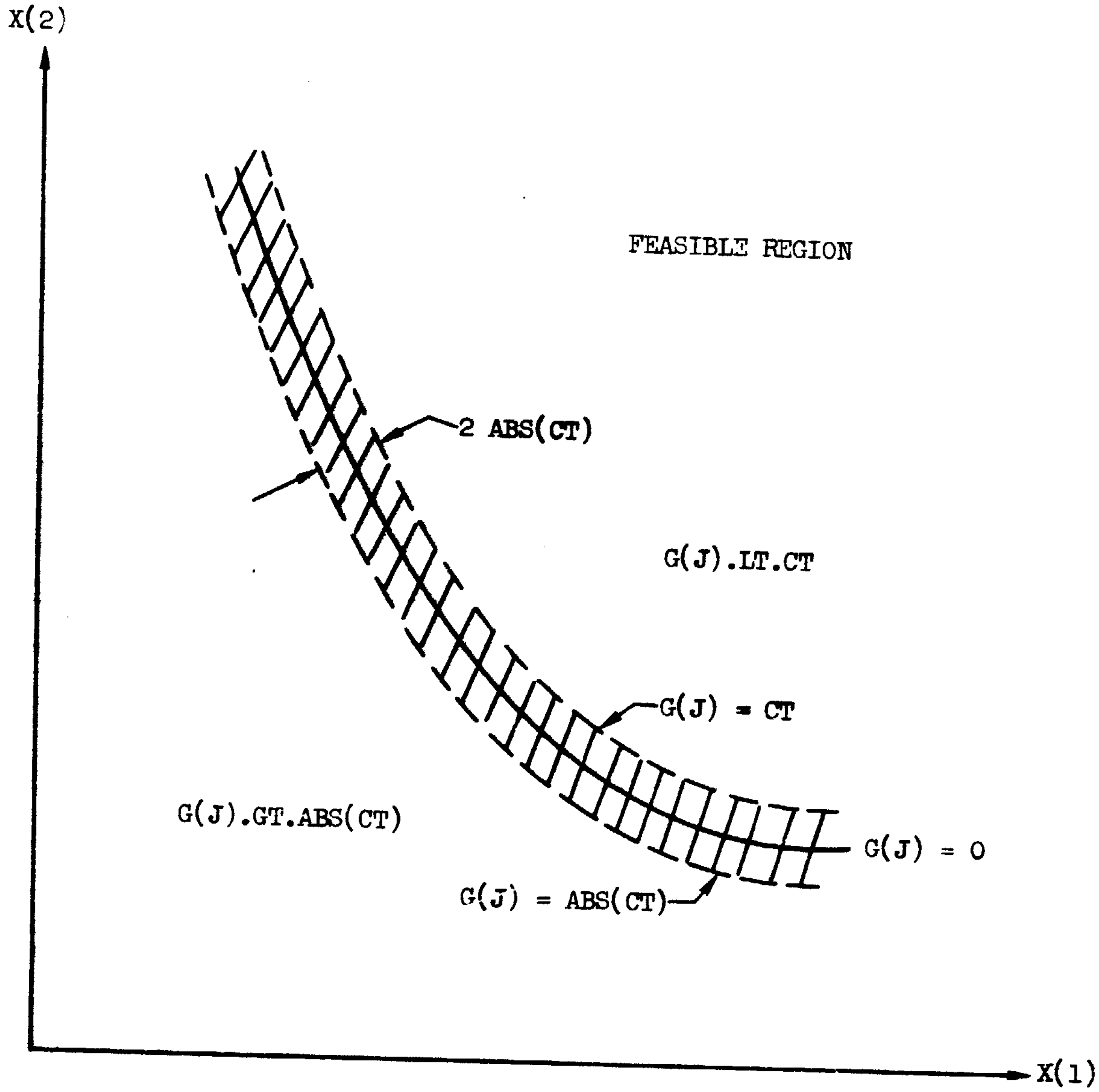


FIG. 1 - CONSTRAINT THICKNESS PARAMETER, CT.

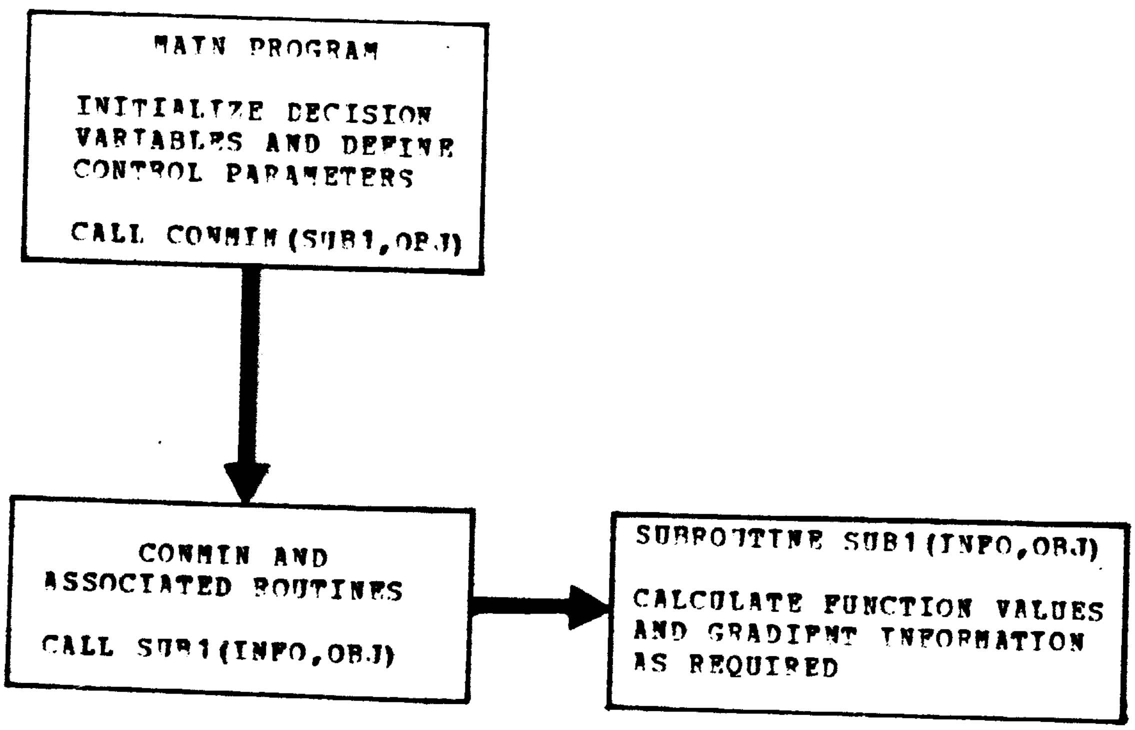


FIG. 2 - PROGRAM ORGANIZATION.

```
SUBROUTINE SUB1(INFO,OBJ)
 COMMON /CNMN1/ IPRINT,NDV,ITMAX,NCON,NSIDE,ICNDIR,NSCAL,NFDG,
1 FDCH,FDCHM,CT,CTMIN,CTL,CTLMIN,THETA,PHI,NACMX1,DELFUN,
2 DABFUN,LINOBJ,ITRM,ITER,NCAL(4)
 COMMON /CNMN2/ X(N1),DF(N1),G(N2),ISC(N8),IC(N4),A(N4,N3)
 COMMON /CNMN3/ S(N3),G1(N7),G2(N2),C(N9),MS1(N6),B(N4,N4)
 COMMON /CNMN4/ VLB(N1),VUB(N1),SCAL(N5)
```

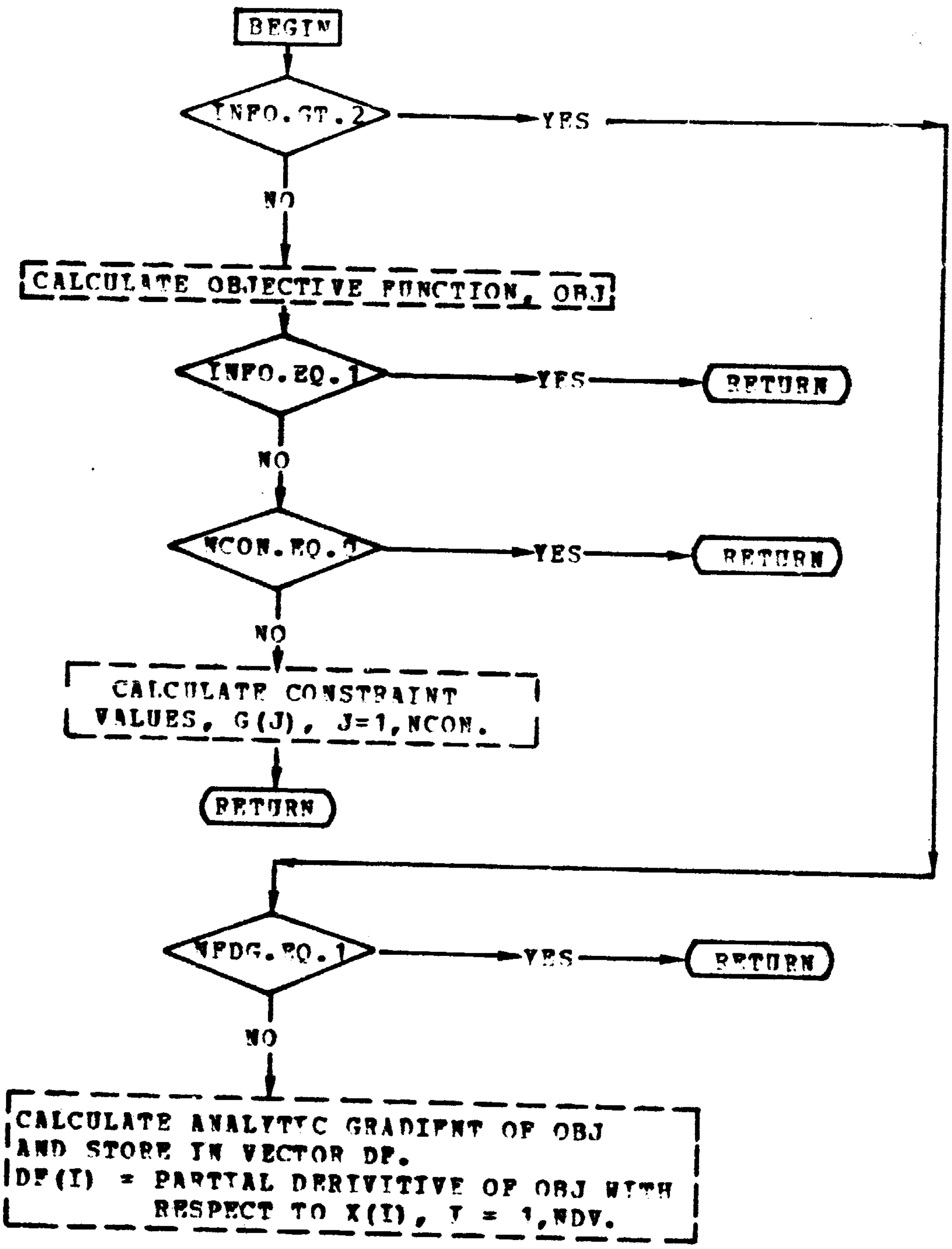


FIG. 3 - ORGANIZATION OF EXTERNAL ROUTINE SUB1

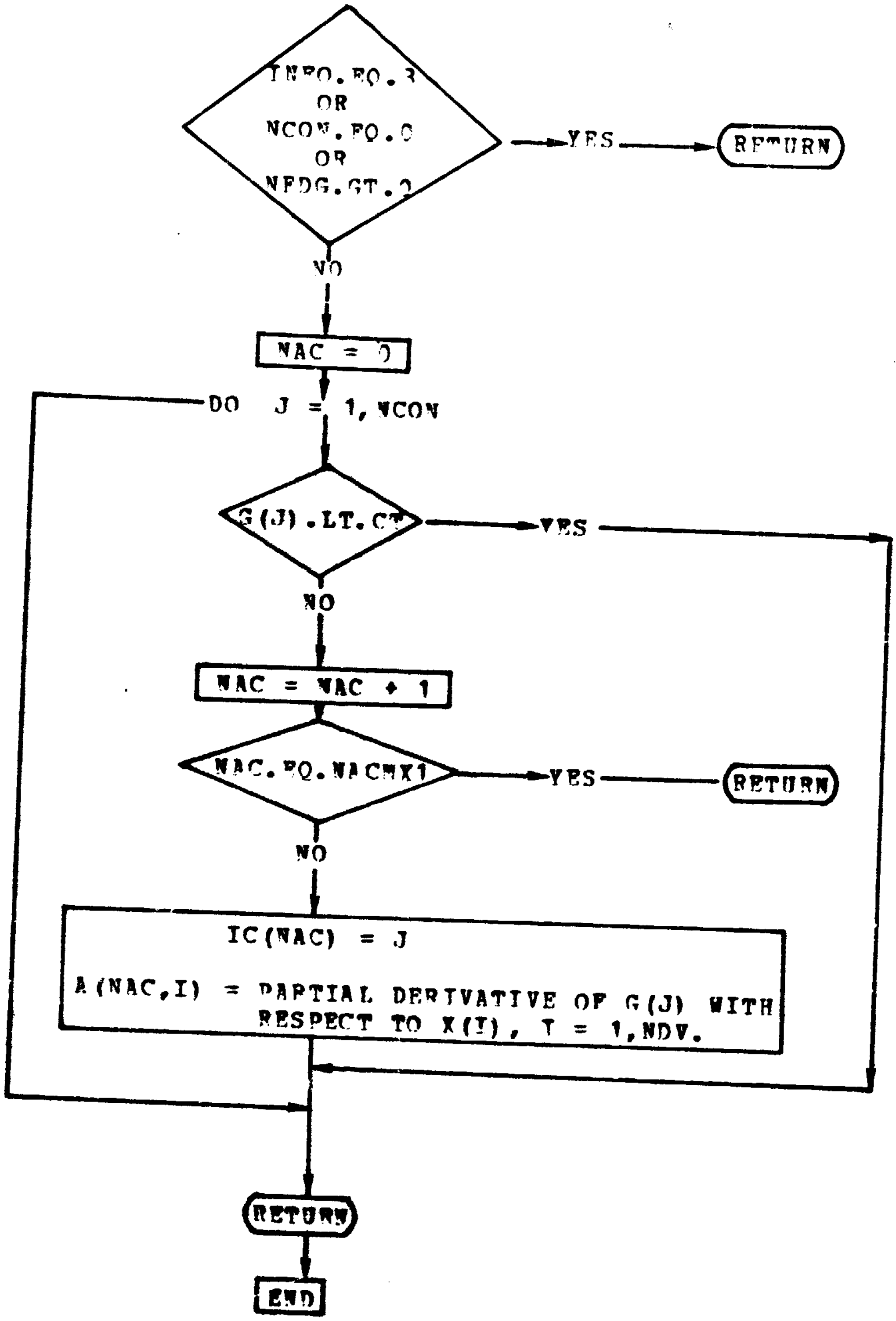


FIG. 3 - ORGANIZATION OF EXTERNAL ROUTINE SUB1 - CONT.

```
C     EXAMPLES 1 AND 2 OF CONMIN USER'S MANUAL.
C     MINIMIZATION OF CONSTRAINED ROSEN-SUZUKI FUNCTION.
      COMMON /CNMN1/ IPRINT,NDV,ITMAX,NCON,NSIDE,ICNDIR,NSCAL,NFDG,
     1 FDCH,FDCHM,CT,CTMIN,CTL,CTLMIN,THETA,PHI,NAC,NACMX1,DELFUN,
     2 DABFUN,LINOBJ,ITRM,ITER,NCAL(4)
      COMMON /CNMN2/ X(50),DF(50),G(1100),ISC(1000),IC(51),A(51,52)
      COMMON /CNMN4/ VLB(50),VUB(50),SCAL(50)
C     DEFINE NAMES OF EXTERNAL ROUTINE, SUB1.
      EXTERNAL EXMP1,EXMP2
C     INITIALIZE CONTROL PARAMETERS.
C     FOR EXAMPLE 2, CHANGE THE FOLLOWING STATEMENT TO;
C     IPRINT=1
      IPRINT=2
      NDV=4
      ITMAX=40
      NCON=3
C     FOR EXAMPLE 2, CHANGE THE FOLLOWING STATEMENT TO;
C     NFDG=0
      NFDG=1
      NACMX1=51
      NSIDE=0
      ICNDIR=0
      NSCAL=0
      LINOBJ=0
      ITRM=0
      FDCH=0.
      FDCHM=0.
      CT=0.
      CTMIN=0.
      CTL=0.
      CTLMIN=0.
      THETA=0.
      PHI=0.
      DELFUN=0.
      DABFUN=0.
C     INITIALIZE CONSTRAINT IDENTIFICATION VECTOR, ISC.
      DO 10 J=1,NCON
   10 ISC(I)=0
C     INITIALIZE X-VECTOR.
      DO 20 I=1,NDV
   20 X(I)=1.
C     SOLVE OPTIMIZATION.
C     FOR EXAMPLE 2, CHANGE THE FOLLOWING STATEMENT TO;
C     CALL CONMIN(EXMP2,OBJ)
      CALL CONMIN(EXMP1,OBJ)
      STOP
      END
```

FIG. 4 - MAIN PROGRAM FOR EXAMPLES 1 AND 2.

```
      SUBROUTINE EXMP1 (INFO,OBJ)
      COMMON /CNMN2/ X(50),DF(50),G(1100),ISC(1000),IC(51),A(51,52)
C     ROUTINE TO PROVIDE FUNCTION AND CONSTRAINT VALUES FOR
C     OPTIMIZATION OF CONSTRAINED ROSEN-SUZUKI FUNCTION.
C     EXAMPLE 1 OF CONMIN USER'S MANUAL.
C     FUNCTION VALUE.
      OBJ=X(1)**2-5.*X(1)+X(2)**2-5.*X(2)+2.*X(3)**2-21.*X(3)+
     1X(4)**2+7.*X(4)+50.
      IF (INFO.EQ.1) RETURN
C     CONSTRAINT VALUES.
      G(1)=X(1)**2+X(1)+X(2)**2-X(2)+X(3)**2+X(3)+X(4)**2-X(4)-8.
      G(2)=X(1)**2-X(1)+2.*X(2)**2+X(3)**2+2.*X(4)**2-X(4)-10.
      G(3)=2.*X(1)**2+2.*X(1)+X(2)**2-X(2)+X(3)**2-X(4)-5.
      RETURN
      END
```

FIG. 5 - EXTERNAL ROUTINE, EXMP1, FOR EXAMPLE 1.

```
* * * * * * * * * * * * * * * * * * * * * * * * * * * * *
*                                                       *
*                    C O N M I N                        *
*                                                       *
*                FORTRAN PROGRAM FOR                    *
*                                                       *
*         CONSTRAINED FUNCTION MINIMIZATION             *
*                                                       *
* NASA/AMES RESEARCH CENTER, MOFFETT FIELD, CALIF.      *
*                                                       *
* * * * * * * * * * * * * * * * * * * * * * * * * * * * *
```

```
CONSTRAINED FUNCTION MINIMIZATION

CONTROL PARAMETERS

IPRINT  NDV    ITMAX   NCON    NSIDE   ICNDIR  NSCAL   NFDG
  2      4      40      3       0        5       0       1

NACMX1  LINOBJ  ITRM
 51      0       3

FDCH             FDCHM
 1.00000E-02     1.00000E-02

CT               CTMIN            CTL              CTLMIN
-1.00000E-01      4.00000E-03     -1.00000E-02      1.00000E-03

THETA            PHI              DELFUN           DABFUN
 1.00000E+00      2.50000E+01      1.00000E-04      3.10000E-02

INITIAL FUNCTION INFORMATION

OBJ =    3.100000E+01

X-VECTOR
 1.0000E+00    1.0000E+00    1.0000E+00    1.0000E+00

CONSTRAINT VALUES
-4.0000E+00   -6.0000E+00   -1.0000E+00

THERE ARE      0 ACTIVE CONSTRAINTS AND      0 VIOLATED CONSTRAINTS
```

FIG. 6 - OPTIMIZATION RESULTS FOR EXAMPLE 1.

```
ITER =    1     OBJ =   2.54843E+01

X-VECTOR
 1.0436E+00   1.0436E+00   1.2479E+00   8.6847E-01


ITER =    2     OBJ =   1.22043E+01

X-VECTOR
-6.5498E-01   1.0325E+00   2.3572E+00   1.3804E-01


ITER =    3     OBJ =   8.37629E+00

X-VECTOR
 2.2440E-01   9.9268E-01   2.0345E+00  -3.1841E-01


ITER =    4     OBJ =   6.94203E+00

X-VECTOR
-3.4392E-01   1.0043E+00   2.1498E+00  -8.0388E-01


ITER =    5     OBJ =   6.32708E+00

X-VECTOR
-6.7566E-02   1.0136E+00   2.0734E+00  -8.1323E-01


ITER =    6     OBJ =   6.17226E+00

X-VECTOR
-9.4581E-02   9.9247E-01   2.0400E+00  -9.6346E-01


ITER =    7     OBJ =   6.07060E+00

X-VECTOR
 7.4640E-02   9.8928E-01   1.9478E+00  -1.0562E+00
```

FIG. 6 - OPTIMIZATION RESULTS FOR EXAMPLE 1 - CONT.

```
ITER =    8     OBJ =   6.02184E+00
X-VECTOR
-1.7653E-02   1.0038E+00   2.0139E+00  -9.7523E-01


ITER =    9     OBJ =   6.01829E+00
X-VECTOR
 1.2500E-02   9.9690E-01   1.9943E+00  -1.0006E+00


ITER =   10     OBJ =   6.01829E+00     NO CHANGE IN OBJ


ITER =   11     OBJ =   6.01829E+00     NO CHANGE IN OBJ
```

FIG. 6 - OPTIMIZATION RESULTS FOR EXAMPLE 1 - CONT.

```
FINAL OPTIMIZATION INFORMATION

 OBJ =    6.018287E+00

 X-VECTOR
  1.2500E-02   9.9690E-01   1.9943E+00  -1.0006E+00

 CONSTRAINT VALUES
 -1.7146E-02  -1.0445E+00  -2.8422E-14

 ACTIVE CONSTRAINT NUMBERS
 NO. GREATER THAN    3  INDICATES SIDE CONSTRAINT
     3

 TERMINATION CRITERION
      ABS(OBJ(I)-OBJ(I-1))   LESS THAN DABFUN FOR  3 ITERATIONS

 NUMBER OF ITERATIONS =   11

 OBJECTIVE FUNCTION WAS EVALUATED           68   TIMES

 CONSTRAINT FUNCTIONS WERE EVALUATED        64   TIMES
```

FIG. 6 - OPTIMIZATION RESULTS FOR EXAMPLE 1 - CONT.

```
      SUBROUTINE EXMP2 (INFO,OBJ)
      COMMON /CNMN1/ IPRINT,NDV,ITMAX,NCON,NSIDE,ICNDIR,NSCAL,NFDG,
     1 FDCH,FDCHM,CT,CTMIN,CTL,CTLMIN,THETA,PHI,NAC,NACMX1,DELFUN,
     2 DABFUN,LINOBJ,ITRM,ITER,NCAL(4)
      COMMON /CNMN2/ X(50),DF(50),G(1100),ISC(1000),IC(51),A(51,50)
C     ROUTINE TO PROVIDE FUNCTION AND CONSTRAINT VALUES FOR
C     OPTIMIZATION OF CONSTRAINED ROSEN-SUZUKI FUNCTION.
C     EXAMPLE 2 OF CONMIN USER'S MANUAL.
      IF (INFO.GT.2) GO TO 10
C     FUNCTION VALUE.
      OBJ=X(1)**2-5.*X(1)+X(2)**2-5.*X(2)+2.*X(3)**2-21.*X(3)+
     1X(4)**2+7.*X(4)+50.
      IF (INFO.EQ.1) RETURN
C     CONSTRAINT VALUES.
      G(1)=X(1)**2+X(1)+X(2)**2-X(2)+X(3)**2+X(3)+X(4)**2-X(4)-8.
      G(2)=X(1)**2-X(1)+2.*X(2)**2+X(3)**2+2.*X(4)**2-X(4)-10.
      G(3)=2.*X(1)**2+2.*X(1)+X(2)**2-X(2)+X(3)**2-X(4)-5.
      RETURN
10    CONTINUE
C     GRADIENT OF OBJECTIVE FUNCTION.
      DF(1)=1.*X(1)-5.
      DF(2)=2.*X(2)-5.
      DF(3)=4.*X(3)-21.
      DF(4)=2.*X(4)+7.
      IF (INFO.EQ.3) RETURN
C     GRADIENTS OF ACTIVE AND VIOLATED CONSTRAINTS.
      NAC=0
      IF (G(1).LT.CT) GO TO 20
      NAC=1
      IC(1)=1
      A(1,1)=2.*X(1)+1.
      A(1,2)=2.*X(2)-1.
      A(1,3)=2.*X(3)+1.
      A(1,4)=2.*X(4)-1.
20    IF (G(2).LT.CT) GO TO 30
      NAC=NAC+1
      IF (NAC.EQ.NACMX1) RETURN
      IC(NAC)=2
      A(NAC,1)=2.*X(1)-1.
      A(NAC,2)=4.*X(2)
      A(NAC,3)=2.*X(3)
      A(NAC,4)=4.*X(4)-1.
30    IF (G(3).LT.CT) RETURN
      NAC=NAC+1
      IF (NAC.EQ.NACMX1) RETURN
      IC(NAC)=3
      A(NAC,1)=4.*X(1)+2.
      A(NAC,2)=2.*X(2)-1.
      A(NAC,3)=2.*X(3)
      A(NAC,4)=-1.
      RETURN
      END
```

FIG. 7 - EXTERNAL ROUTINE, EXMP2, FOR EXAMPLE 2.

```
* * * * * * * * * * * * * * * * * * * * * * * * * * * *
*                                                      *
*                    C O N M I N                       *
*                                                      *
*                FORTRAN PROGRAM FOR                   *
*                                                      *
*        CONSTRAINED FUNCTION MINIMIZATION             *
*                                                      *
*  NASA/AMES RESEARCH CENTER, MOFFETT FIELD, CALIF.    *
*                                                      *
* * * * * * * * * * * * * * * * * * * * * * * * * * * *

INITIAL FUNCTION INFORMATION

OBJ =    3.100000E+01

X-VECTOR
 1.0000E+00   1.0000E+00   1.0000E+00   1.0000E+00

CONSTRAINT VALUES
-4.0000E+00  -6.0000E+00  -1.0000E+00

THERE ARE      0 ACTIVE CONSTRAINTS AND      0 VIOLATED CONSTRAINTS
```

FIG. 8 - OPTIMIZATION RESULTS FOR EXAMPLE 2.

```
FINAL OPTIMIZATION INFORMATION

 OBJ =    6.006646E+00

 X-VECTOR
 -7.1710E-03   1.0022E+00   2.0053E+00  -9.9073E-01

 CONSTRAINT VALUES
 -6.0054E-03  -1.0089E+00  -8.5265E-14

 ACTIVE CONSTRAINT NUMBERS
 NO. GREATER THAN    3  INDICATES SIDE CONSTRAINT
     1    3

 TERMINATION CRITERION
      ABS(OBJ(I)-OBJ(I-1))   LESS THAN DABFUN FOR  3 ITERATIONS

 NUMBER OF ITERATIONS =   13

 OBJECTIVE FUNCTION WAS EVALUATED             34  TIMES

 CONSTRAINT FUNCTIONS WERE EVALUATED          34  TIMES

 GRADIENT OF OBJECTIVE WAS CALCULATED         12  TIMES

 GRADIENTS OF CONSTRAINTS WERE CALCULATED     12  TIMES
```

FIG. 8 -OPTIMIZATION RESULTS FOR EXAMPLE 2 - CONT.

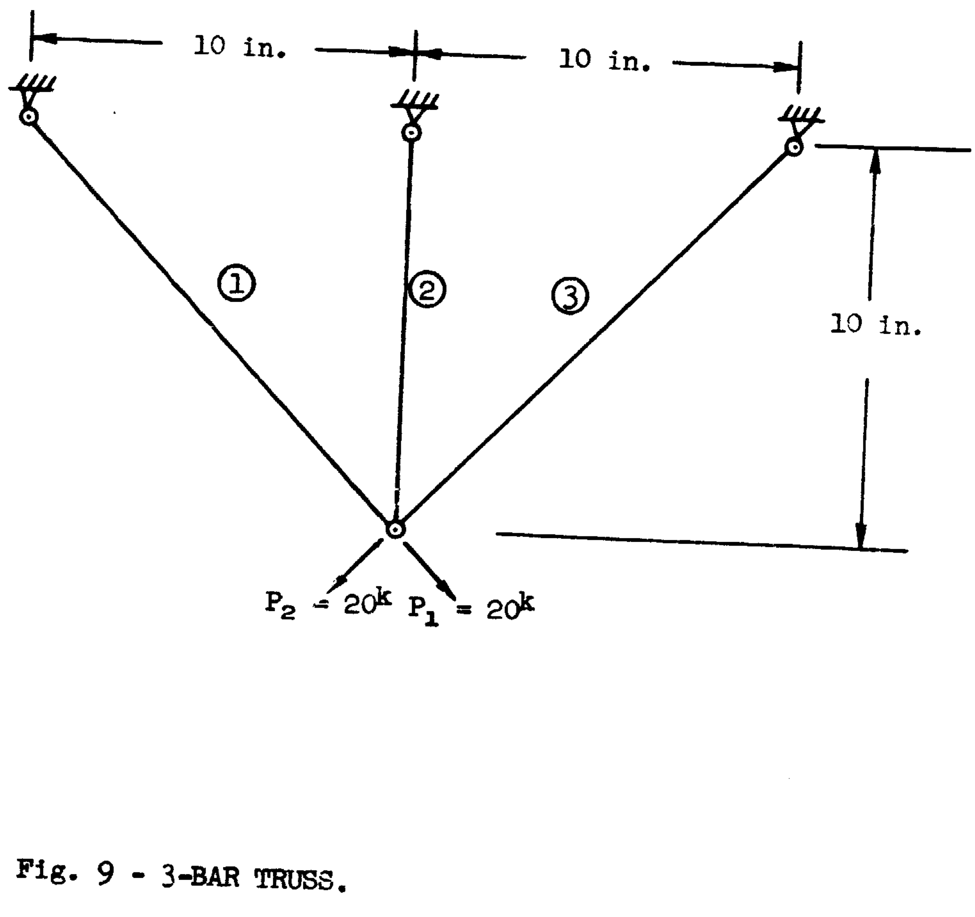


Fig. 9 - 3-BAR TRUSS.

```
C     EXAMPLE 3 OF CONMIN USER'S MANUAL.
C     MINIMIZATION OF WEIGHT OF THE 3-BAR TRUSS.
      COMMON /CNMN1/ IPRINT,NDV,ITMAX,NCON,NSIDE,ICNDIR,NSCAL,NFDG,
     1 FDCH,FDCHM,CT,CTMIN,CTL,CTLMIN,THETA,PHI,NAC,NACMX1,DELFUN,
     2 DABFUN,LINOBJ,ITRM,ITER,NCAL(4)
      COMMON /CNMN2/ X(50),DF(50),G(1100),ISC(1000),IC(51),A(51,52)
      COMMON /CNMN4/ VLB(50),VUB(50),SCAL(50)
C     DEFINE NAME OF EXTERNAL ROUTINE, SUB1.
      EXTERNAL EXMP3
C     INITIALIZE CONTROL PARAMETERS.
      IPRINT=3
      NDV=2
      NCON=6
      NSIDE=1
      NFDG=2
      NACMX1=51
      LINOBJ=1
      ITMAX=0
      ICNDIR=0
      NSCAL=0
      ITRM=0
      FDCH=0.
      FDCHM=0.
      CT=0.
      CTMIN=0.
      CTL=0.
      CTLMIN=0.
      THETA=0.
      PHI=0.
      DELFUN=0.
      DABFUN=0.
C     INITIALIZE CONSTRAINT IDENTIFICATION VECTOR, ISC.
      DO 10 J=1,NCON
10    ISC(I)=0
C     INITIALIZE LOWER AND UPPER BOUNDING VECTORS.
      DO 20 I=1,NDV
      VLB(I)=00.001
20    VUB(I)=1.0E+10
C     INITIALIZE X-VECTOR.
      DO 30 I=1,NDV
30    X(I)=1.
C     SOLVE OPTIMIZATION.
      CALL CONMIN(EXMP3,OBJ)
      STOP
      END
```

FIG. 10 - MAIN PROGRAM FOR EXAMPLE 3.

```
      SUBROUTINE EXMP3 (INFO,OBJ)
      COMMON /CNMN2/ X(50),DF(50),G(1100),ISC(1000),IC(51),A(51,52)
C     ROUTINE TO PROVIDE FUNCTION AND GRADIENT INFORMATION FOR
C     3-BAR TRUSS.
C     EXAMPLE 3 OF CONMIN USER'S MANUAL.
      RHO=0.1
      A1=X(1)
      A2=X(2)
      IF (INFO.GT.2) GO TO 10
C     OBJECTIVE FUNCTION.
      OBJ=10.*RHO*(2.*SQRT(2.)*A1+A2)
      IF (INFO.EQ.1) RETURN
C     CONSTRAINT VALUES.
      DENOM=2.*A1*A2+SQRT(2.)*A1*A1
      SIG11=20.*(SQRT(2.)*A1+A2)/DENOM
      SIG21=20.*SQRT(2.)*A1/DENOM
      SIG31=-20.*A2/DENOM
      G(1)=-SIG11/15.-1.
      G(2)=SIG11/20.-1.
      G(3)=-SIG21/15.-1.
      G(4)=SIG21/20.-1.
      G(5)=-SIG31/20.-1.
      G(6)=SIG31/20.-1.
      RETURN
10    CONTINUE
C     GRADIENT OF OBJECTIVE FUNCTION.
      DF(1)=20.*SQRT(2.)*RHO
      DF(2)=10.*RHO
      RETURN
      END
```

FIG. 11 - EXTERNAL ROUTINE, EXMP3, FOR EXAMPLE 3.

```
* * * * * * * * * * * * * * * * * * * * * * * * * * * * *
*                                                       *
*                   C O N M I N                         *
*                                                       *
*              FORTRAN PROGRAM FOR                      *
*                                                       *
*       CONSTRAINED FUNCTION MINIMIZATION               *
*                                                       *
*  NASA/AMES RESEARCH CENTER, MOFFETT FIELD, CALIF.     *
*                                                       *
* * * * * * * * * * * * * * * * * * * * * * * * * * * * *
```

```
CONSTRAINED FUNCTION MINIMIZATION

CONTROL PARAMETERS

IPRINT  NDV   ITMAX  NCON   NSIDE  ICNDIR  NSCAL  NFDG
  3      2     10     6      1       3       0      2

NACMX1  LINOBJ  ITRM
 26       1       3

FDCH            FDCHM
 1.00000E-02    1.00000E-02

CT              CTMIN           CTL             CTLMIN
-1.00000E-01     4.00000E-03    -1.00000E-02     1.00000E-03

THETA           PHI             DELFUN          DABFUN
 1.00000E+00     2.50000E+01     1.00000E-04     3.82843E-03

LOWER BOUNDS ON DECISION VARIABLES
 1.0000E-03   1.0000E-03

UPPER BOUNDS ON DECISION VARIABLES
 1.0000E+10   1.0000E+10


INITIAL FUNCTION INFORMATION

OBJ =   3.828427E+00

X-VECTOR
 1.0000E+00   1.0000E+00

CONSTRAINT VALUES
-1.9428E+00  -2.9289E-01  -1.5523E+00  -5.8579E-01  -6.0948E-01  -1.2929E+00

THERE ARE     0 ACTIVE CONSTRAINTS AND     0 VIOLATED CONSTRAINTS
```

FIG. 12 - OPTIMIZATION RESULTS FOR EXAMPLE 3.

```
BEGIN ITERATION NUMBER     1

S-VECTOR
-2.8284E+00  -1.0000E+00

SLOPE =  -9.00000E+00     ALPHA =   1.14385E-01

OBJ =   2.798966E+00

X-VECTOR
 6.7647E-01   8.8562E-01



BEGIN ITERATION NUMBER     2

CONSTRAINT VALUES
-2.3311E+00  -1.6582E-03  -1.6912E+00  -4.8158E-01  -3.6011E-01  -1.4799E+00

PUSH-OFF FACTORS, THETA(I), I=1,NAC
 9.6711E-01

NO. OF ACTIVE CONSTRAINTS =    1  OF WHICH    0  ARE VIOLATED
    0  ACTIVE OR VIOLATED CONSTRAINTS ARE SIDE CONSTRAINTS

ACTIVE CONSTRAINT NUMBERS
NO. GREATER THAN     6  INDICATES SIDE CONSTRAINT
    2

CONSTRAINT PARAMETER, BETA =   8.16624E-04

S-VECTOR
 2.2667E-02  -9.0398E-02

SLOPE =  -2.62871E-02     ALPHA =   6.44333E+00

OBJ =   2.629589E+00

X-VECTOR
 8.2252E-01   3.0315E-01



BEGIN ITERATION NUMBER     3

CONSTRAINT VALUES
-2.3433E+00   7.4916E-03  -2.0656E+00  -2.0079E-01  -7.2229E-01  -1.2083E+00

PUSH-OFF FACTORS, THETA(I), I=1,NAC
 1.1554E+00
```

FIG. 12 - OPTIMIZATION RESULTS FOR EXAMPLE 3 - CONT.

```
NO. OF ACTIVE CONSTRAINTS =     1  OF WHICH    0  ARE VIOLATED
    0  ACTIVE OR VIOLATED CONSTRAINTS ARE SIDE CONSTRAINTS

ACTIVE CONSTRAINT NUMBERS
NO. GREATER THAN    6  INDICATES SIDE CONSTRAINT

    2

CONSTRAINT PARAMETER, BETA =   2.53279E-05

S-VECTOR
-1.0595E-02   2.7380E-02

SLOPE =  -2.58809E-03     ALPHA =   1.05049E-02

OBJ =   2.629562E+00

X-VECTOR
 8.2241E-01   3.0344E-01



BEGIN ITERATION NUMBER    4

CONSTRAINT VALUES
-2.3433E+00   7.4795E-03  -2.0654E+00  -2.0098E-01  -7.2205E-01  -1.2085E+00

PUSH-OFF FACTORS, THETA(I), I=1,NAC
 1.4851E+00

NO. OF ACTIVE CONSTRAINTS =     1  OF WHICH    0  ARE VIOLATED
    0  ACTIVE OR VIOLATED CONSTRAINTS ARE SIDE CONSTRAINTS

ACTIVE CONSTRAINT NUMBERS
NO. GREATER THAN    6  INDICATES SIDE CONSTRAINT
    2

CONSTRAINT PARAMETER, BETA =   1.63884E-05

S-VECTOR
-9.0708E-03   2.3720E-02

SLOPE =  -1.93600E-03     ALPHA =   4.25485E+00

OBJ =   2.621324E+00

X-VECTOR
 7.8381E-01   4.0436E-01
```

FIG. 12 - OPTIMIZATION RESULTS FOR EXAMPLE 3 - CONT.

```
BEGIN ITERATION NUMBER    5

CONSTRAINT VALUES
-2.3423E+00   6.7277E-03  -1.9835E+00  -2.6236E-01  -6.4122E-01  -1.2691E+00

PUSH-OFF FACTORS, THETA(I), I=1,NAC
 1.4320E+00

NO. OF ACTIVE CONSTRAINTS =     1  OF WHICH    0  ARE VIOLATED
    0  ACTIVE OR VIOLATED CONSTRAINTS ARE SIDE CONSTRAINTS

ACTIVE CONSTRAINT NUMBERS
NO. GREATER THAN    6  INDICATES SIDE CONSTRAINT
    2

CONSTRAINT PARAMETER, BETA =   6.46576E-11


OBJ =   2.621324E+00     NO CHANGE ON OBJ



BEGIN ITERATION NUMBER    6

CONSTRAINT VALUES
-2.3423E+00   6.7277E-03  -1.9835E+00  -2.6236E-01  -6.4122E-01  -1.2691E+00

PUSH-OFF FACTORS, THETA(I), I=1,NAC
 2.4802E+00

NO. OF ACTIVE CONSTRAINTS =     1  OF WHICH    0  ARE VIOLATED
    0  ACTIVE OR VIOLATED CONSTRAINTS ARE SIDE CONSTRAINTS

ACTIVE CONSTRAINT NUMBERS
NO. GREATER THAN    6  INDICATES SIDE CONSTRAINT
    2

CONSTRAINT PARAMETER, BETA =   2.20652E-11

OBJ =   2.621324E+00     NO CHANGE ON OBJ



BEGIN ITERATION NUMBER    7

CONSTRAINT VALUES
-2.3423E+00   6.7277E-03  -1.9835E+00  -2.6236E-01  -6.4122E-01  -1.2691E+00

PUSH-OFF FACTORS, THETA(I), I=1,NAC
 7.1843E+00
```

FIG. 12 - OPTIMIZATION RESULTS FOR EXAMPLE 3 - CONT.

```
NO. OF ACTIVE CONSTRAINTS =     1  OF WHICH     1  ARE VIOLATED
    0  ACTIVE OR VIOLATED CONSTRAINTS ARE SIDE CONSTRAINTS

ACTIVE CONSTRAINT NUMBERS
NO. GREATER THAN     6  INDICATES SIDE CONSTRAINT
    2

CONSTRAINT PARAMETER, BETA =   7.71561E+01

S-VECTOR
 2.2189E+01   7.8705E+00

SLOPE =    7.06315E+01      ALPHA =   2.49714E-04

OBJ =    2.638962E+00

X-VECTOR
 7.8936E-01   4.0633E-01



BEGIN ITERATION NUMBER    8

CONSTRAINT VALUES
-2.3333E+00  -4.2505E-11  -1.9775E+00  -2.6686E-01  -6.4419E-01  -1.2669E+00

PUSH-OFF FACTORS, THETA(I), I=1,NAC
 1.0000E+00

NO. OF ACTIVE CONSTRAINTS =     1  OF WHICH     0  ARE VIOLATED
    0  ACTIVE OR VIOLATED CONSTRAINTS ARE SIDE CONSTRAINTS

ACTIVE CONSTRAINT NUMBERS
NO. GREATER THAN     6  INDICATES SIDE CONSTRAINT
    2

CONSTRAINT PARAMETER, BETA =   3.49062E-10

OBJ =    2.638962E+00      NO CHANGE ON OBJ
```

FIG. 12 - OPTIMIZATION RESULTS FOR EXAMPLE 3 - CONT.

```
FINAL OPTIMIZATION INFORMATION

 OBJ =   2.638962E+00

 X-VECTOR
  7.8936E-01   4.0633E-01

 CONSTRAINT VALUES
 -2.3333E+00  -4.2505E-11  -1.9775E+00  -2.6686E-01  -6.4419E-01  -1.2669E+00

 ACTIVE CONSTRAINT NUMBERS
 NO. GREATER THAN    6  INDICATES SIDE CONSTRAINT
     2

 TERMINATION CRITERION
      ABS(CT) IS LESS THAN CTMIN

 NUMBER OF ITERATIONS =    8

 OBJECTIVE FUNCTION WAS EVALUATED             26  TIMES

 CONSTRAINT FUNCTIONS WERE EVALUATED          26  TIMES

 GRADIENT OF OBJECTIVE WAS CALCULATED          6  TIMES
```

FIG. 12 - OPTIMIZATION RESULTS FOR EXAMPLE 3 - CONT.

APPENDIX A
SUMMARY OF PARAMETERS USED BY CONMIN

COMMON BLOCKS:

```
 COMMON /CNMN1/ IPRINT,NDV,ITMAX,NCON,NSIDE,ICNDIR,NSCAL,NFDG,
1 FDCH,FDCHM,CT,CTMIN,CTL,CTLMIN,THETA,PHI,NACMX1,DELFUN,
2 DABFUN,LINOBJ,ITRM,ITER,NCAL(4)
 COMMON /CNMN2/ X(N1),DF(N1),G(N2),ISC(N8),IC(N4),A(N4,N3)
 COMMON /CNMN3/ S(N3),G1(N7),G2(N2),C(N9),MS1(N6),B(N4,N4)
 COMMON /CNMN4/ VLB(N1),VUB(N1),SCAL(N5)
```

CALL STATEMENTS:

```
CALL CONMIN(SUB1,OBJ)
CALL(SUB1(INFO,OBJ)
```

PARAMETERS DEFINED IN THE MAIN PROGRAM.

PARAM.	DEFAULT	DEFINITION
IPRINT		Print control.
NDV		Number of decision variables, X(I).
ITMAX	10	Maximum number of iterations in the minimization process.
NCON		Number of constraint functions, G(J).
NSIDE		Side constraint parameter. NSIDE.GT.0 indicates that lower and upper bounds are imposed on the decision variables.
ICNDIR	NDV+1	Conjugate direction restart parameter. Restart with steepest descent move every ICNDIR iterations.
NSCAL		Scaling control parameter. NSCAL.LT.0, user supplies scaling vector. NSCAL.EQ.0, no scaling. NSCAL.GT.0, automatic linear scaling every nNSCAL iterations.

PARAM.	DEFAULT	DEFINITION
NFDG		Gradient calculation control parameter.
FDCH	0.01	Relative change in decision variable, X(I), in calculating finite difference gradients.
FDCHM	0.01	Minimum absolute step in finite difference gradient calculations.
CT	-0.1	Constraint thickness parameter.
CTMIN	0.004	Minimum absolute value of CT considered in optimization process.
CTL	-0.01	Constraint thickness parameter for linear and side constraints.
CTLMIN	0.001	Minimum absolute value of CTL considered in optimization process.
THETA	1.0	Mean value of push-off factor in method of feasible directions.
PHI	5.0	Participation coefficient, used if one or more constraints are violated.
NACMX1		1 plus user's best estimate of the maximum number of constraints (including side constraints) which will be active or violated at any time in the minimization process.
DELFUN	0.001	Minimum relative change in objective function, OBJ, to indicate convergence.

PARAM.	DEFAULT	DEFINITION
DABFUN	0.001* Init. OBJ	Minimum absolute change in objective function, OBJ, to indicate convergence.
LINOBJ		Linear objective function identifier. LINOBJ=1 if OBJ is specifically known to be linear in X(I). LINOBJ=0 if OBJ is non-linear.
ITRM	3	Number of consecutive iterations to indicate convergence by relative or absolute changes, DELFUN or DABFUN.
X(N1)		Vector of decision variables.
VLB(N1)		Vector of lower bounds on decision variables.
VUB(N1)		Vector of upper bounds on decision variables.
SCAL(N5)		Vector of scaling parameters.
ISC(N8)		Linear constraint identification vector.

PARAMETERS DEFINED IN EXTERNAL ROUTINE SUB1

PARAMETER	DEFINITION
OBJ	Value of objective function.
G(N2)	Vector of constraint values.
DF(N1)	Analytic gradient of objective function.
NAC	Number of active and violated constraints (G(J).GE.CT).
A(N4,N3)	Row I contains analytic gradient of the Ith active or violated constraint.
IC(N4)	Identifies which constraints are active or violated.

PARAMETERS DEFINED IN CONMIN AND ASSOCIATED ROUTINES

PARAMETER	DEFINITION
ITER	Iteration number.
NCAL(4)	Bookkeeping information. NCAL(I) gives number of times INFO=I during optimization process.
S(N3)	Direction vector.
G1(N7)	Temporary storage of vectors G and X.
G(N2)	Temporary storage of vector G.
B(N4,N4)	Used in finding usable-feasible direction.
C(N9)	Used in finding usable-feasible direction and for temporary storage of vector X.

APPENDIX B

CONMIN SUPROUTINE DESCRIPTIONS

Following is a list of the subroutines associated with CONMIN. If the array dimensions are changed from those currently used, the common blocks in each routine must be changed accordingly.

CONMIN - Main optimization routine.

CNMN01 - Routine to calculate gradient information by finite difference.

CNMN02 - Calculate direction of steepest descent, or conjugate direction in unconstrained function minimization.

CNMN03 - Solve one-dimensional search in unconstrained function minimization.

CNMN04 - Find minimum of one-dimensional function by polynomial interpolation.

CNMN05 - Determine usable-feasible, or modified usable-feasible, direction in constrained function minimization.

CNMN06 - Solve one-dimensional search for constrained function minimization.

CNMN07 - Find zero of one-dimensional function by polynomial interpolation.

CNMN08 - Solve special linear programming problem in determination of usable-feasible, or modified usable-feasible direction in constrained function minimization.

CNMN09 - Unscale and rescale decision variables before and after function evaluation.

REFERENCES

1. Lund, S., "Direct Search Program for Optimization of Non-linear Functions with Non-linear Constraints", Users Manual SK/P13, NTH, Trondheim, Norway, 1971.

2. Zoutendijk, G., "Methods of Feasible Directions", Elsevier Publishing Co., Amsterdam, 1960.

3. Vanderplaats, G. N., and Moses, F., "Structural Optimization by Methods of Feasible Directions", National Symposium on Computerized Structural Analysis and Design, Washington, D. C., March, 1972.

4. Fletcher, R. and Reeves, C. M., "Function Minimization by Conjugate Gradients", British Computer J., Vol. 7, No. 2, 1964.

5. Schmit, L. A., Jr., "Structural Design by Systematic Synthesis", Proceedings of the Second National Conference on Electronic Computation, Structural Division, ASCE, Pittsburgh, Pa., Sept., 1960.

END

DATE

FILMED

OCT 4 1973