# SUMMARY

CONMIN is a FORTRAN program, in subroutine form, for the solution of linear or nonlinear 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 describes 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.


# 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 referring 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 curve in design space. However, on a digital computer, the exact value of G(J) = 0 can seldom be obtained. Therefore, the "curve" becomes a thick band with constraint thickness of 2*ABS(CT) over which G(J) is assumed to be zero. Because all G(J) must be negative, CT is taken as a negative number for consistency 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*
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 description of the subroutines associated with CONMIN.

# 2 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 nonlinear 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 source code, 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 nonlinear.
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.

# 3 PROGRAM ORGANIZATION

Since the original release of the CONMIN program, numerous modifications and improvements have been made. These include changes to the program organization, COMMON block structures, array dimensions and control parameters. An understanding of these changes is necessary for the operation of the current version of the program and these are outlined in the following sections.

This version of CONMIN is identified by a comment card near the beginning of subroutine CONMIN:

C * * MAY, 1978 VERSION * * .

## PROGRAM ORGANIZATION

The original version of CONMIN was written such that a user-supplied subroutine was called by CONMIN for function and gradient calculations. The current version is organized such that the function evaluation routine is contained in, or is called by, the main program. CONMIN executes according to the parameter IGOTO which must be initialized to zero. Figure A1 shows the required program organization. The purpose of this new logic is so that the program can be used in an overlay system or can be restarted in mid-execution.

## COMMON BLOCKS

CNMN2, CNMN3 and CNMN4 as originally defined are no longer required. However, the arrays contained in these common blocks must still be dimensioned in the main program. Common block name CONSAV is now used for internal storage by CONMIN and this name must not be used elsewhere. The information in common block CNMN1 has been changed, and is now:

COMMON /CNMN1/ DELFUN, DABFUN, FDCH, FDCHM, CT, CTMIN, CTL, CTLMIN, ALPHAX, ABOBJ1, THETA, obj, NDV, NCON, NSIDE, IPRINT, NFDG, NSCAL, LINOBJ, ITMAX, ITRM, ICNDIR, IGOTO, nac, info, infog, iter.

NOTE: The parameters typed in upper case must be initialized before the first call to CONMIN. The parameters typed in bold characters are calculated by the user during execution, depending on the value of info; nac will be calculated by the user only if NFDG = 1.

## ARRAY DIMENSIONS

The storage requirements for the arrays used in CONMIN have changed. The current storage requirements are:

```
DIMENSION X(N1), VLB(N1), VUB(N1), g(N2), SCAL(N1), df(N1), a(N1,N3), s(N1), g1(N2), g2
```

where

```
N1 = NDV + 2
N2 = NCON + 2*NDV
N3 = NACMX1
N4 = MAX (N3,NDV)
N5 = 2*N4
```

Note that these are minimum dimensions. The arrays may be dimensioned larger than this at the user's option.

NOTE: Uppercase denotes arrays which must be initialized before the first call to CONMIN. VLB, VUB, ISC and SCAL must be initialized only if they will be used: VLB and VUB if NSIDE.GT.0, ISC if NCON.GT.0 and SCAL if NSCAL.LT.0.

During execution, the arrays g, a, and isc must be calculated by the user, depending on the value of NFDG. If NFDG = 0, only array g is calculated. If NFDG = 2, only arrays g and df are calculated. The remaining arrays are used internally by CONMIN.

## CALLING STATEMENT

CONMIN is now called by the FORTRAN statement:

```
CALL CONMIN (X, VLB, VUB, G, SCAL, DF, A, S, G1, G2, B, C, ISC, IC, MS1, N1, N2, N3, N4
```

CAUTION: Check that the values of N1, N2, N3, N4 and N5 in the calling statement are consistent with the array dimensions discussed above.

## PARAMETER CHANGES

Variable NACMX1 defined in common block CNMN1 is replaced by N3 in the parameter list.

A new parameter, INFOG, has been added:

INFOG = 0: same as when INFOG was not used.

INFOG = 1: only those constraints identified as active or violated in array IC(I), I = 1, NAC need be evaluated. This is only meaningful if finite difference gradients are calculated, and allows the user to avoid calculating non-essential information. If it is convenient to evaluate all constraints each time, variable INFOG may be ignored.

The parameters ALPHAX and ABOBJ1 have been added to CNMN1.

ALPHAX (default = 0.1) is the maximum fractional change in any component of X as an initial estimate for ALPHA in the one-dimensional search. That is, the initial ALPHA will be such that no component of X is changed by more than this amount. This only applies to those X(i) of magnitude greater than 0.1. If an optimization run shows numerous ALPHA = 0 results for the one-dimensional search, it may help to try ALPHAX less than the default. ALPHAX is changed by CONMIN depending on the progress of the optimization.

ABOBJ1 (default = 0.1) is the fractional change attempted as a first step in the one-dimensional search and is based on a linear approximation. ABOBJ1 is updated during the optimization, depending on progress. The initial step in the one-dimensional search is taken as the amount necessary to change OBJ by ABOBJ1*ABS(OBJ) or to change some X(i) by ALPHAX*ABS( X(i) ), whichever is less.

The definition of parameter INFO has changed. The current definition is:

INFO = 1: calculate OBJ and G(I), I = 1, NCON

INFO = 2: calculate NAC, IC(I), I = 1,NAC, the gradient of OBJ, and the gradient of G(J), where J = IC(I), I = 1,NAC. Store the gradients of G in columns of A.

The definition of the finite difference gradient parameter, NFDG has changed. The current definition is:

NFDG = 0: all gradient information is calculated by finite difference within CONMIN.

NFDG = 1: all gradient information is supplied by the user.

NFDG = 2: the gradient of OBJ is supplied by the user and the gradients of constraints are calculated by finite difference within CONMIN.

Additional printing is now available using the IPRINT parameter:

IPRINT = 1, 2, 3, 4: same as before.

IPRINT = 5: all of above plus each proposed design vector, objective and constraints during the one-dimensional search.

### GRADIENT STORAGE

The gradients of active or violated constraints are now stored in the columns of array [A], instead of the rows of [A]. This is for computational efficiency and convenience. Also, it sometimes simplifies programming associated with user-supplied gradients (NFDG = 1). This change only effects those users who use the NFDG = 1 option. If gradients of constraints are calculated by CONMIN the user only needs to be sure that array [A] is correctly dimensioned in the calling program.

### OVERLAY/RESTART CAPABILITY

CONMIN can now be in its own overlay. It is only required that the arrays be saved in overlay zero as well as the contents of common blocks CNMN1 and CONSAV. CONSAV contains 50 real parameters followed by 25 integer parameters:

```
COMMON /CONSAV/ REAL(50), INT(25)
```

This also allows the user to restart CONMIN at any point during the optimization. Upon return from CONMIN it is only necessary to write the information in the parameter list (all arrays plus N1 - N5) on disc, together with the contents of CNMN1 and CONSAV. The program can be restarted by reading this information back from disc and continuing the program execution from this point.

### DOUBLE PRECISION OPERATIONS

Occasionally it is desirable to use CONMIN in double precision, usually on an IBM machine. To do this, add implicit READ*8 A-H, O-Z cards at the beginning of each routine.

NOTE: This has not been tested because on unavailability of an IBM computer, but care has been taken to insure that no parity errors will occur. If you use double precision please inform the author of success or failure so other users can be assured of the operational status of this option.

## 4 PARAMETERS DEFINED IN MAIN PROGRAM

IPRINT Print control. All printing is done on unit 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 (i.e., 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 if 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 zigzagging 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 nonlinear, 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 deri- vative 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.

# 5 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 NFDG = 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.

# 6 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.

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.

# 7 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 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*
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. 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
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 analysis subroutine ANALYS are listed in Listing 1 and Listing 2 respectively, with the optimization results in Listing 3. An optimum design of OBJ = 6.01 is obtained with the corresponding decision variables:

```
X = (0.0194, 0.995, 1.99, -1.01)
```

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 for NFDG and IPRINT. The gradient of the objective function with respect to the decision variables is:

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

The gradients of the constraint functions are:

```
| 2*X(1) + 1 |                           | 2*X(1) - 1 |
grad(G(1)) =  | 2*X(2) - 1 |        grad(G(2)) = | 4*X(2)     |
              | 2*X(3) + 1 |                     | 2*X(3)     |
              | 2*X(4) - 1 |                     | 4*X(4) - 1 |
              | 4*X(1) + 2 |
grad(G(3)) =  | 2*X(2) - 1 |
              | 2*X(3)     |
              |   - 1      |
```

The main program is the same as before (Listing 1). The subroutine is in Listing 4 and the optimization results in Listing 5, where an optimum design of OBJ = 6.01 is obtained with

```
X = (0.027, 0.995, 1.98, -1.01)
```

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 the gradient of OBJ is defined by:

```
               | 20.0*SQRT(2.0)*RHO |
grad(OBJ)  =   |                    |
               | 10.0*RHO           |
```

The print control will be taken as IPRINT = 1 and the default values are used for all other control parameters. Then the control parameters are defined as:

```
IPRINT = 1, NDV = 2, NCON = 6, NSIDE = 1, NFDG = 2, LINOBJ = 1,
ITMAX = ICNDIR = NSCAL = ITRM = DABFUN = 0,
FDCH = FDCHM = CT = CTMIN = CTL = CTLMIN = THETA = PHI = DELFUN = 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 and analysis subroutine for this example is in Listing 6 The optimization results are given in Listing 7, where:

```
OBJ = 2.63
```

and

```
X = (0.78, 0.43)
```

Note that only constraint number 2 (the tensile stress constraint in member 1 under load condition 1) is active. These results were produced on a MicroVAX workstation using single precision arithmetic.

# 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, 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)
```

CALL STATEMENTS:

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

PARAMETERS DEFINED IN THE MAIN PROGRAM:

| PARAMETERS | 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 NSCAL iterations. |
| 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. |
| DABFUN | 0.001*initial 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 nonlinear. |
| ITRM | 3 | Number of consecutive iterations to indicate convergence by relative or absolute changes, DELFUN or DABFUN. |
| X | | Vector of decision variables. |
| VLB | | Vector of lower bounds on decision variables. |
| VUB | | Vector of upper bounds on decision variables. |
| SCAL | | Vector of scaling parameters. |
| ISC | | Linear constraint identification vector. |

PARAMETERS DEFINED IN EXTERNAL ROUTINE SUB1:

| PARAMETER | DEFINITION |
|---|---|
| OBJ | Value of objective function. |
| G | Vector of constraint values. |
| DF | Analytic gradient of objective function. |
| NAC | Number of active and violated constraints (G(J).GE.CT). |
| A | Matrix containing analytic gradients of active or violated constraints. |
| IC | 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 that INFO = I during optimization process. |
| S | Direction vector. |
| G1 | Temporary storage of vectors G and X. |
| G2 | Temporary storage of vector G. |
| B | Used in finding usable-feasible direction. |
| C | Used in finding usable-feasible direction and for temporary storage of vector X. |

# B CONMIN SUBROUTINE 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.

# C LISTING 1: MAIN PROGRAM FOR EXAMPLES 1 & 2

```
CCCCC
      PROGRAM EXAMPL1
      DIMENSION S(6),G1(11),G2(11),B(11,11),C(11),MS1(22)
      DIMENSION VLB(6),VUB(6),SCAL(6),DF(6),A(6,11),
     .              ISC(11),IC(11)
      COMMON /CNMN1/ DELFUN,DABFUN,FDCH,FDCHM,CT,CTMIN,CTL,CTLMIN,
     .               ALPHAX,ABOBJ1,THETA,OBJ,NDV,NCON,NSIDE,IPRINT,
     .               NFDG,NSCAL,LINOBJ,ITMAX,ITRM,ICNDIR,IGOTO,NAC,
     .               INFO,INFOG,ITER
      COMMON /VARABLE/ AOBJ,X(6),G(11)
      COMMON /ANDATA/ LOOPCNT
      NAMELIST /CONPAR/ INFOG,INFO,NFDG,IPRINT,NDV,ITMAX,NCON,NSIDE,
```

```
     .                    ICNDIR,NSCAL,FDCH,FDCHM,CT,CTMIN,CTLMIN,THETA,
     .                    PHI,DELFUN,DABFUN,LINOBJ,ITRM,X,VLB,VUB,
     .                    N1,N2,N3,N4,N5,ALPHAX,ABOBJ1,CTL,ISC,SCAL
C
C     THIS PROGRAM EXECUTES THE EXAMPLE PROBLEM ONE OF THE CONMIN
C     MANUAL.
C
C
C  INITIALIZE
C
      INFOG=0
      INFO=0
      NFDG=0
      IPRINT=2
      NDV=4
      ITMAX=40
      NCON=3
      NSIDE=0
      ICNDIR=0
      NSCAL=0
      FDCH=0.0
      FDCHM=0.0
      CT=0.0
      CTMIN=0.0
      CTL=0.0
      CTLMIN=0.0
      THETA=0.0
      PHI=0.0
      DELFUN=0.0
      DABFUN=0.0
      LINOBJ=0.0
      ITRM=0
      N1=6
      N2=11
      N3=11
      N4=11
      N5=22
      ALPHAX=0.0
      ABOBJ1=0.0
      CTL=0.0
      DO 5 I=1,NDV
        X(I)=1.0
        VLB(I)=-99999.
        VUB(I)= 99999.
    5 CONTINUE
C
```

```
      DO 6 J=1,NCON
        ISC(J)=0
    6 CONTINUE
C
C     READ THE PARAMETERS FOR CONMIN
C
CCC   READ(5,CONPAR)                    USE DEFAULT VALUES
      WRITE(6,CONPAR)
      NLIM=ITMAX*(NDV+5)
C
C     NON-ITERATIVE PART OF ANALYSIS
C
      IGOTO = 0
C
C     ITERATIVE PART OF ANALYSIS
C
      DO 1000 I = 1,NLIM
        LOOPCNT=I
C
C       CALL THE OPTIMIZATION ROUTINE CONMIN
C
        CALL CONMIN(X,VLB,VUB,G,SCAL,DF,A,S,G1,G2,B,C,ISC,IC,MS1,N1,N2,
     .              N3,N4,N5)
C
        IF(IGOTO.EQ.0) LOOPCNT=-999
C
C       ANALYSIS MODULE
C
        CALL ANALYS
        OBJ=AOBJ
        IF (IGOTO.EQ.0) GO TO 1100
 1000 CONTINUE
C
C
 1100 CONTINUE
      STOP
      END
CCCCC
```

# D LISTING 2: ANALYSIS SUBROUTINE FOR EXAMPLE 1

```
CCCCC
      SUBROUTINE ANALYS
```

```
      COMMON /VARABLE/ AOBJ,X(6),G(11)
C
C   ROUTINE TO CALCULATE OBJECTIVE FUNCTION AND
C   CONSTRAINTS
C
C
C  OBJECTIVE FUNCTION
C
      AOBJ = X(1)**2 - 5.*X(1) + X(2)**2 - 5.*X(2) + 2.*X(3)**2
     .       - 21.*X(3) + X(4)**2 + 7.0*X(4) + 50.
C
C
C   CONSTRAINT VALUES
C
      G(1) = X(1)**2 + X(1) + X(2)**2 - X(2) + X(3)**2 + X(3)
     .       + X(4)**2 - X(4) - 8.0
C
      G(2) = X(1)**2 - X(1) + 2. * X(2)**2 + X(3)**2 + 2.*X(4)**2
     .       - X(4) - 10.0
C
      G(3) = 2.*X(1)**2 + 2.*X(1) + X(2)**2 - X(2) + X(3)**2 - X(4) -5.0
C
      RETURN
      END
CCCCC
```

# E LISTING 3: OPTIMIZATION RESULTS FOR EXAMPLE 1

```
1$CONPAR
INFOG   = 0,
INFO    = 0,
NFDG    = 0,
IPRINT  = 2,
NDV     = 4,
ITMAX   = 40,
NCON    = 3,
NSIDE   = 0,
ICNDIR  = 0,
NSCAL   = 0,
FDCH    = 0.0,
FDCHM   = 0.0,
CT      = 0.0,
CTMIN   = 0.0,
```

```
CTLMIN  = 0.0,
THETA   = 0.0,
PHI     = 0.0,
DELFUN  = 0.0,
DABFUN  = 0.0,
LINOBJ  = 0,
ITRM    = 0,
X       = .1E+01, .1E+01, .1E+01, .1E+01, 0.0, 0.0,
VLB     = -.99999E+05, -.99999E+05, -.99999E+05, -.99999E+05, 0.0, 0.0,
VUB     = .99999E+05, .99999E+05, .99999E+05, .99999E+05, 0.0, 0.0,
N1      = 6,
N2      = 11,
N3      = 11,
N4      = 11,
N5      = 22,
ALPHAX  = 0.0,
ABOBJ1  = 0.0,
CTL     = 0.0,
ISC     = 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
SCAL    = 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
$END

            * * * * * * * * * * * * * * * * * * * * * * * * * * *
            *                                                   *
            *                   C O N M I N                     *
            *                                                   *
            *               FORTRAN PROGRAM FOR                 *
            *                                                   *
            *         CONSTRAINED FUNCTION MINIMIZATION         *
            *                                                   *
            * * * * * * * * * * * * * * * * * * * * * * * * * * *



     CONSTRAINED FUNCTION MINIMIZATION

     CONTROL PARAMETERS

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

     LINOBJ  ITRM     N1      N2      N3      N4      N5
       0       3       6      11      11      11      22

         CT               CTMIN           CTL              CTLMIN
```

```
 -.10000E+00      .40000E-02     -.10000E-01      .10000E-02

    THETA            PHI             DELFUN          DABFUN
   .10000E+01      .50000E+01      .10000E-03      .31000E-01

    FDCH            FDCHM            ALPHAX          ABOBJ1
   .10000E-01      .10000E-01      .10000E+00      .10000E+00

ALL CONSTRAINTS ARE NON-LINEAR


INITIAL FUNCTION INFORMATION

OBJ =    .310000E+02

DECISION VARIABLES (X-VECTOR)
  1)     .10000E+01   .10000E+01   .10000E+01   .10000E+01

CONSTRAINT VALUES (G-VECTOR)
  1)    -.40000E+01  -.60000E+01  -.10000E+01




ITER =    1     OBJ =    .25484E+02

DECISION VARIABLES (X-VECTOR)
  1)     .10436E+01   .10436E+01   .12479E+01   .86847E+00

CONSTRAINT VALUES (G-VECTOR)
  1)    -.31307E+01  -.55788E+01  -.56843E-13




ITER =    2     OBJ =    .12204E+02

DECISION VARIABLES (X-VECTOR)
  1)    -.65498E+00   .10325E+01   .23572E+01   .13804E+00

CONSTRAINT VALUES (G-VECTOR)
  1)    -.39775E+00  -.13275E+01  -.76739E-12
```

```
ITER =    3     OBJ =    .83763E+01

DECISION VARIABLES (X-VECTOR)
  1)     .22440E+00   .99268E+00   .20345E+01  -.31841E+00

CONSTRAINT VALUES (G-VECTOR)
  1)    -.11388E+01  -.35427E+01  -.85265E-13



ITER =    4     OBJ =    .69420E+01

DECISION VARIABLES (X-VECTOR)
  1)    -.34392E+00   .10043E+01   .21498E+01  -.80388E+00

CONSTRAINT VALUES (G-VECTOR)
  1)    -.11369E-12  -.80266E+00  -.21613E-01



ITER =    5     OBJ =    .63271E+01

DECISION VARIABLES (X-VECTOR)
  1)    -.67566E-01   .10136E+01   .20734E+01  -.81323E+00

CONSTRAINT VALUES (G-VECTOR)
  1)    -.20225E+00  -.14382E+01   .28422E-13



ITER =    6     OBJ =    .61723E+01

DECISION VARIABLES (X-VECTOR)
  1)    -.94581E-01   .99247E+00   .20400E+01  -.96346E+00

CONSTRAINT VALUES (G-VECTOR)
  1)    -.56843E-13  -.94507E+00  -.53852E-01



ITER =    7     OBJ =    .60706E+01
```

```
DECISION VARIABLES (X-VECTOR)
  1)     .74640E-01   .98928E+00   .19478E+01  -.10562E+01

CONSTRAINT VALUES (G-VECTOR)
  1)    -.16766E-01  -.10302E+01  -.28422E-13



ITER =    8     OBJ =    .60218E+01

DECISION VARIABLES (X-VECTOR)
  1)    -.17653E-01   .10038E+01   .20139E+01  -.97523E+00

CONSTRAINT VALUES (G-VECTOR)
  1)    -.17726E-01  -.10338E+01  0.



ITER =    9     OBJ =    .60182E+01

DECISION VARIABLES (X-VECTOR)
  1)     .23921E-01   .99428E+00   .19869E+01  -.10102E+01

CONSTRAINT VALUES (G-VECTOR)
  1)    -.15891E-01  -.10472E+01   .11747E-02



ITER =   10     OBJ =    .60133E+01

DECISION VARIABLES (X-VECTOR)
  1)    -.17147E-01   .10055E+01   .20139E+01  -.97533E+00

CONSTRAINT VALUES (G-VECTOR)
  1)    -.15050E-01  -.10270E+01   .29211E-02



ITER =   11     OBJ =    .60098E+01

DECISION VARIABLES (X-VECTOR)
```

```
  1)     .19441E-01   .99482E+00   .19908E+01  -.10058E+01

 CONSTRAINT VALUES (G-VECTOR)
  1)    -.13894E-01  -.10474E+01   .34703E-02


FINAL OPTIMIZATION INFORMATION

 OBJ =    .600982E+01

 DECISION VARIABLES (X-VECTOR)
  1)     .19441E-01   .99482E+00   .19908E+01  -.10058E+01

 CONSTRAINT VALUES (G-VECTOR)
  1)    -.13894E-01  -.10474E+01   .34703E-02

 THERE ARE    2 ACTIVE CONSTRAINTS
 CONSTRAINT NUMBERS ARE
     1    3

 THERE ARE    0 VIOLATED CONSTRAINTS

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

 NUMBER OF ITERATIONS =   11

 OBJECTIVE FUNCTION WAS EVALUATED          78  TIMES

 CONSTRAINT FUNCTIONS WERE EVALUATED       78  TIMES
```

# F LISTING 4: ANALYSIS SUBROUTINE FOR EXAMPLE 2

```
CCCCC
      PROGRAM EXAMPL2
      DIMENSION S(6),G1(11),G2(11),B(11,11),C(11),MS1(22)
      DIMENSION VLB(6),VUB(6),SCAL(6)
      COMMON/GRAD/ ISC(11),IC(11),DF(6),A(6,11)
      COMMON /CNMN1/ DELFUN,DABFUN,FDCH,FDCHM,CT,CTMIN,CTL,CTLMIN,
     .               ALPHAX,ABOBJ1,THETA,OBJ,NDV,NCON,NSIDE,IPRINT,
     .               NFDG,NSCAL,LINOBJ,ITMAX,ITRM,ICNDIR,IGOTO,NAC,
     .               INFO,INFOG,ITER
      COMMON /VARABLE/ AOBJ,X(6),G(11)
```

```
      COMMON /ANDATA/ LOOPCNT
      NAMELIST /CONPAR/ INFOG,INFO,NFDG,IPRINT,NDV,ITMAX,NCON,NSIDE,
     .                  ICNDIR,NSCAL,FDCH,FDCHM,CT,CTMIN,CTLMIN,THETA,
     .                  PHI,DELFUN,DABFUN,LINOBJ,ITRM,X,VLB,VUB,
     .                  N1,N2,N3,N4,N5,ALPHAX,ABOBJ1,CTL,ISC,SCAL
C
C     THIS PROGRAM EXECUTES THE EXAMPLE PROBLEM TWO OF THE CONMIN MANUAL.
      OPEN( UNIT=6,FILE='EXOUT2.TXT',STATUS='NEW')
C
C
C  INITIALIZE
C
      INFOG=0
      INFO=0
      NFDG=1
      IPRINT=1
      NDV=4
      ITMAX=40
      NCON=3
      NSIDE=0
      ICNDIR=0
      NSCAL=0
      FDCH=0.0
      FDCHM=0.0
      CT=0.0
      CTMIN=0.0
      CTL=0.0
      CTLMIN=0.0
      THETA=0.0
      PHI=0.0
      DELFUN=0.0
      DABFUN=0.0
      LINOBJ=0.0
      ITRM=0
      N1=6
      N2=11
      N3=11
      N4=11
      N5=22
      ALPHAX=0.0
      ABOBJ1=0.0
      CTL=0.0
      DO 5 I=1,NDV
        X(I)=1.0
        VLB(I)=-99999.
        VUB(I)= 99999.
```

```
    5 CONTINUE
C
      DO 6 J=1,NCON
        ISC(J)=0
    6 CONTINUE
C
C     READ THE PARAMETERS FOR CONMIN
C
CCC   READ(5,CONPAR)      USE DEFAULT VALUES
      WRITE(6,CONPAR)
      NLIM=ITMAX*(NDV+5)
C
C     NON-ITERATIVE PART OF ANALYSIS
C
      IGOTO = 0
C
C     ITERATIVE PART OF ANALYSIS
C
      DO 1000 I = 1,NLIM
        LOOPCNT=I
C
C       CALL THE OPTIMIZATION ROUTINE CONMIN
C
        CALL CONMIN(X,VLB,VUB,G,SCAL,DF,A,S,G1,G2,B,C,ISC,IC,MS1,N1,N2,
     .              N3,N4,N5)
C
        IF(IGOTO.EQ.0) LOOPCNT=-999
C
C       ANALYSIS MODULE
C
        CALL ANALYS
        OBJ=AOBJ
        IF (IGOTO.EQ.0) GO TO 1100
 1000 CONTINUE
C
C
 1100 CONTINUE
      STOP
      END
      SUBROUTINE ANALYS
      COMMON /VARABLE/ AOBJ,X(6),G(11)
      COMMON/GRAD/ ISC(11),IC(11),DF(6),A(6,11)
      COMMON /CNMN1/ DELFUN,DABFUN,FDCH,FDCHM,CT,CTMIN,CTL,CTLMIN,
     .               ALPHAX,ABOBJ1,THETA,OBJ,NDV,NCON,NSIDE,IPRINT,
     .               NFDG,NSCAL,LINOBJ,ITMAX,ITRM,ICNDIR,IGOTO,NAC,
     .               INFO,INFOG,ITER
```

```
C
C   ROUTINE TO CALCULATE OBJECTIVE FUNCTION AND
C   CONSTRAINT  VALUES FOR OPTIMIZATION OF CONSTRAINED ROSEN-SUZUKI
C   FUNCTION.
C
C
      IF(INFO.GE.2) GO TO 10
C
C  OBJECTIVE FUNCTION
C
      AOBJ = X(1)**2 - 5.*X(1) + X(2)**2 - 5.*X(2) + 2.*X(3)**2
     .       - 21.*X(3) + X(4)**2 + 7.0*X(4) + 50.
C
C
C   CONSTRAINT VALUES
C
      G(1) = X(1)**2 + X(1) + X(2)**2 - X(2) + X(3)**2 + X(3)
     .       + X(4)**2 - X(4) - 8.0
C
      G(2) = X(1)**2 - X(1) + 2. * X(2)**2 + X(3)**2 + 2.*X(4)**2
     .       - X(4) - 10.0
C
      G(3) = 2.*X(1)**2 + 2.*X(1) + X(2)**2 - X(2) + X(3)**2 - X(4) -5.0
C
      GO TO 999
   10 CONTINUE
C
C
C    GRADIENT INFORMATION
C
      DF(1)=2.0*X(1) - 5.0
      DF(2)=2.0*X(2) - 5.0
      DF(3)=4.0*X(3) - 21.
      DF(4)=2.0*X(4) + 7.
C
C  GRADIENTS OF ACTIVE AND VIOLATED CONSTRAINTS
C
      NAC=0
      IF(G(1).LT.CT) GO TO 20
      NAC=1
      IC(1)=1
      A(1,1)=2.*X(1)+1.
      A(2,1)=2.*X(2)-1.
      A(3,1)=2.*X(3)+1.
      A(4,1)=2.*X(4)-1.
C
```

```
   20 IF(G(2).LT.CT) GO TO 30
      NAC=NAC+1
      IC(NAC)=2
      A(1,NAC)=2.*X(1)-1.0
      A(2,NAC)=4.*X(2)
      A(3,NAC)=2.*X(3)
      A(4,NAC)=4.*X(4)-1.0
C
   30 IF(G(3).LT.CT) GO TO 999
      NAC=NAC+1
      IC(NAC)=3
      A(1,NAC)=4.*X(1)+2.
      A(2,NAC)=2.*X(2)-1.
      A(3,NAC)=2.*X(3)
      A(4,NAC)=-1.
  999 RETURN
      END
CCCCC
```

# G LISTING 5: OPTIMIZATION RESULTS FOR EXAMPLE 2

```
$CONPAR
 INFOG   =          0,
 INFO    =          0,
 NFDG    =          1,
 IPRINT  =          1,
 NDV     =          4,
 ITMAX   =         40,
 NCON    =          3,
 NSIDE   =          0,
 ICNDIR  =          0,
 NSCAL   =          0,
 FDCH    =  0.0000000E+00,
 FDCHM   =  0.0000000E+00,
 CT      =  0.0000000E+00,
 CTMIN   =  0.0000000E+00,
 CTLMIN  =  0.0000000E+00,
 THETA   =  0.0000000E+00,
 PHI     =  0.0000000E+00,
 DELFUN  =  0.0000000E+00,
 DABFUN  =  0.0000000E+00,
 LINOBJ  =          0,
 ITRM    =          0,
```

```
X       = 4*1.000000    , 2*0.0000000E+00,
VLB     = 4*-99999.00    , 2*0.0000000E+00,
VUB     = 4*99999.00    , 2*0.0000000E+00,
N1      =           6,
N2      =          11,
N3      =          11,
N4      =          11,
N5      =          22,
ALPHAX  =  0.0000000E+00,
ABOBJ1  =  0.0000000E+00,
CTL     =  0.0000000E+00,
ISC     = 11*0,
SCAL    = 6*0.0000000E+00
$END
           * * * * * * * * * * * * * * * * * * * * * * * * * * *
           *                                                   *
           *                   C O N M I N                     *
           *                                                   *
           *               FORTRAN PROGRAM FOR                 *
           *                                                   *
           *         CONSTRAINED FUNCTION MINIMIZATION         *
           *                                                   *
           * * * * * * * * * * * * * * * * * * * * * * * * * * *
    INITIAL FUNCTION INFORMATION
    OBJ =   0.310000E+02
    DECISION VARIABLES (X-VECTOR)
      1)    0.10000E+01  0.10000E+01  0.10000E+01  0.10000E+01
    CONSTRAINT VALUES (G-VECTOR)
      1)   -0.40000E+01 -0.60000E+01 -0.10000E+01
   FINAL OPTIMIZATION INFORMATION
    OBJ =   0.601078E+01
    DECISION VARIABLES (X-VECTOR)
      1)    0.26916E-01  0.99458E+00  0.19848E+01 -0.10128E+01
    CONSTRAINT VALUES (G-VECTOR)
      1)   -0.14837E-01 -0.10438E+01  0.21458E-02
    THERE ARE    2 ACTIVE CONSTRAINTS
    CONSTRAINT NUMBERS ARE
        1    3
    THERE ARE    0 VIOLATED CONSTRAINTS
    TERMINATION CRITERION
         ABS(OBJ(I)-OBJ(I-1))   LESS THAN DABFUN FOR  3 ITERATIONS
    NUMBER OF ITERATIONS =   11
    OBJECTIVE FUNCTION WAS EVALUATED           34  TIMES
    CONSTRAINT FUNCTIONS WERE EVALUATED        34  TIMES
    GRADIENT OF OBJECTIVE WAS CALCULATED       11  TIMES
    GRADIENTS OF CONSTRAINTS WERE CALCULATED   11  TIMES
```

# H LISTING 6: MAIN PROGRAM AND ANALYSIS SUBROUTINE FOR EXAMPLE 3

```
ccccc
      PROGRAM EXAMPL3
      DIMENSION S(4),G1(10),G2(10),B(10,10),C(10),MS1(20)
      DIMENSION VLB(4),VUB(4),SCAL(4)
      COMMON /VARABLE/ AOBJ,X(4),G(10)
      COMMON/GRAD/ ISC(10),IC(10),DF(4),A(4,10)
      COMMON/CONSAV/ RNUM(50),INUM(25)
      COMMON /CNMN1/ DELFUN,DABFUN,FDCH,FDCHM,CT,CTMIN,CTL,CTLMIN,
     .               ALPHAX,ABOBJ1,THETA,OBJ,NDV,NCON,NSIDE,IPRINT,
     .               NFDG,NSCAL,LINOBJ,ITMAX,ITRM,ICNDIR,IGOTO,NAC,
     .               INFO,INFOG,ITER
      COMMON /ANDATA/ LOOPCNT
      NAMELIST /CONPAR/ INFOG,INFO,NFDG,IPRINT,NDV,ITMAX,NCON,NSIDE,
     .                  ICNDIR,NSCAL,FDCH,FDCHM,CT,CTMIN,CTLMIN,THETA,
     .                  PHI,DELFUN,DABFUN,LINOBJ,ITRM,X,VLB,VUB,
     .                  N1,N2,N3,N4,N5,ALPHAX,ABOBJ1,CTL,ISC,SCAL
C
      OPEN(UNIT=6,FILE='EXOUT3.TXT',STATUS='NEW')
C
C     THIS PROGRAM EXECUTES THE EXAMPLE PROBLEM THREE OF THE CONMIN MANUAL.
C
C
C  INITIALIZE
C
      INFOG=0
      INFO=0
      NFDG=2
      IPRINT=1
      NDV=2
      ITMAX=40
      NCON=6
      NSIDE=1
      ICNDIR=0
      NSCAL=0
      FDCH=0.0
      FDCHM=0.0
      CT=0.0
      CTMIN=0.0
      CTL=0.0
      CTLMIN=0.0
      THETA=0.0
      PHI=0.0
```

```
      DELFUN=0.0
      DABFUN=0.0
      LINOBJ=1
      ITRM=0
      N1=4
      N2=10
      N3=10
      N4=10
      N5=20
      ALPHAX=0.0
      ABOBJ1=0.0
      CTL=0.0
      DO 5 I=1,NDV
        X(I)=1.0
        VLB(I)=0.001
        VUB(I)= 1.0E+10
    5 CONTINUE
C
      DO 6 J=1,NCON
        ISC(J)=0
    6 CONTINUE
C
C     READ THE PARAMETERS FOR CONMIN
C
CCC   READ(5,CONPAR)      USE DEFAULT VALUES
      WRITE(6,CONPAR)
      NLIM=ITMAX*(NDV+5)
C
C     NON-ITERATIVE PART OF ANALYSIS
C
      IGOTO = 0
C
C     ITERATIVE PART OF ANALYSIS
C
      DO 1000 I = 1,NLIM
        LOOPCNT=I
C
C       CALL THE OPTIMIZATION ROUTINE CONMIN
C
        CALL CONMIN(X,VLB,VUB,G,SCAL,DF,A,S,G1,G2,B,C,ISC,IC,MS1,N1,N2,
     .              N3,N4,N5)
C
        IF(IGOTO.EQ.0) LOOPCNT=-999
C
C       ANALYSIS MODULE
C
```

```
        CALL ANALYS
        OBJ=AOBJ
        IF (IGOTO.EQ.0) GO TO 1100
 1000 CONTINUE
C
C
 1100 CONTINUE
      CLOSE(6)
      STOP
      END
      SUBROUTINE ANALYS
      COMMON /VARABLE/ AOBJ,X(4),G(10)
      COMMON/GRAD/ ISC(10),IC(10),DF(4),A(4,10)
      COMMON /CNMN1/ DELFUN,DABFUN,FDCH,FDCHM,CT,CTMIN,CTL,CTLMIN,
     .               ALPHAX,ABOBJ1,THETA,OBJ,NDV,NCON,NSIDE,IPRINT,
     .               NFDG,NSCAL,LINOBJ,ITMAX,ITRM,ICNDIR,IGOTO,NAC,
     .               INFO,INFOG,ITER
C
C   ROUTINE TO CALCULATE OBJECTIVE FUNCTION AND
C   CONSTRAINT  VALUES FOR OPTIMIZATION OF CONSTRAINED ROSEN-SUZUKI
C   FUNCTION.
C
C
      RHO= 0.1
      A1= X(1)
      A2= X(2)
C
      IF(INFO.GE.2) GO TO 10
C
C  OBJECTIVE FUNCTION
C
      AOBJ= 10.*RHO*(2.*SQRT(2.)*A1+A2)
C
C
C   CONSTRAINT VALUES
C
      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
C
      G(1)= -SIG11/15.-1.
      G(2)= SIG11/20.-1.
      G(3)= -SIG21/15.-1.
      G(4)= SIG21/20.-1.
      G(5)= -SIG31/15.-1.
```

```
      G(6)= SIG31/20.-1.
C
      GO TO 999
   10 CONTINUE
C
C
C    GRADIENT INFORMATION
C
      DF(1)=20.*SQRT(2.)*RHO
      DF(2)=10.*RHO
C
C  GRADIENTS OF ACTIVE AND VIOLATED CONSTRAINTS
C  WILL BE CALCULATED BY FINITE DIFFERENCE WITHIN
C  CONMIN
C
C
  999 RETURN
      END
ccccc
```

# I LISTING 7: OPTIMIZATION RESULTS FOR EXAMPLE 3

```
 $CONPAR
 INFOG   =          0,
 INFO    =          0,
 NFDG    =          2,
 IPRINT  =          1,
 NDV     =          2,
 ITMAX   =         40,
 NCON    =          6,
 NSIDE   =          1,
 ICNDIR  =          0,
 NSCAL   =          0,
 FDCH    =  0.0000000E+00,
 FDCHM   =  0.0000000E+00,
 CT      =  0.0000000E+00,
 CTMIN   =  0.0000000E+00,
 CTLMIN  =  0.0000000E+00,
 THETA   =  0.0000000E+00,
 PHI     =  0.0000000E+00,
 DELFUN  =  0.0000000E+00,
 DABFUN  =  0.0000000E+00,
 LINOBJ  =          1,
```

```
ITRM    =          0,
X       = 2*1.000000    , 2*0.0000000E+00,
VLB     = 2*1.0000000E-03, 2*0.0000000E+00,
VUB     = 2*1.0000000E+10, 2*0.0000000E+00,
N1      =          4,
N2      =         10,
N3      =         10,
N4      =         10,
N5      =         20,
ALPHAX  =  0.0000000E+00,
ABOBJ1  =  0.0000000E+00,
CTL     =  0.0000000E+00,
ISC     = 10*0,
SCAL    = 4*0.0000000E+00
$END
           * * * * * * * * * * * * * * * * * * * * * * * * * * *
           *                                                   *
           *                   C O N M I N                     *
           *                                                   *
           *               FORTRAN PROGRAM FOR                 *
           *                                                   *
           *          CONSTRAINED FUNCTION MINIMIZATION        *
           *                                                   *
           * * * * * * * * * * * * * * * * * * * * * * * * * * *
    INITIAL FUNCTION INFORMATION
    OBJ =   0.382843E+01
    DECISION VARIABLES (X-VECTOR)
      1)    0.10000E+01  0.10000E+01
    CONSTRAINT VALUES (G-VECTOR)
      1)   -0.19428E+01 -0.29289E+00 -0.15523E+01 -0.58579E+00 -0.60948E+00 -0.12929E+
   FINAL OPTIMIZATION INFORMATION
    OBJ =   0.263281E+01
    DECISION VARIABLES (X-VECTOR)
      1)    0.77853E+00  0.43079E+00
    CONSTRAINT VALUES (G-VECTOR)
      1)   -0.23367E+01  0.25257E-02 -0.19608E+01 -0.27942E+00 -0.62408E+00 -0.12819E+
    THERE ARE    1 ACTIVE CONSTRAINTS
    CONSTRAINT NUMBERS ARE
        2
    THERE ARE    0 VIOLATED CONSTRAINTS
    THERE ARE    0 ACTIVE SIDE CONSTRAINTS
    TERMINATION CRITERION
         ABS(OBJ(I)-OBJ(I-1))   LESS THAN DABFUN FOR  3 ITERATIONS
    NUMBER OF ITERATIONS =    7
    OBJECTIVE FUNCTION WAS EVALUATED          32  TIMES
    CONSTRAINT FUNCTIONS WERE EVALUATED       32  TIMES
```

```
GRADIENT OF OBJECTIVE WAS CALCULATED        6  TIMES
```