NASA TM-87751

NASA Technical Memorandum 87751

SIMULATOR FOR MULTILEVEL OPTIMIZATION RESEARCH

S. L. Padula and K. C. Young

JUNE 1986

NASA
National Aeronautics and
Space Administration

Langley Research Center
Hampton, Virginia 23665

SIMULATOR FOR MULTILEVEL OPTIMIZATION RESEARCH

by S. L. Padula and K. C. Young

Abstract

A computer program designed to simulate and improve multilevel optimization techniques is described. By using simple analytic functions to represent complex engineering analyses, the simulator can generate and test a large variety of multilevel decomposition strategies in a relatively short time. This type of research is an essential step toward routine optimization of large aerospace systems.

The paper discusses the types of optimization problems handled by version 1.0 of the simulator and gives input and output listings and plots for a sample problem. This document serves as a users' manual for the simulator computer program.

Symbols

a,b,c,d	coefficient matrices (see eqs. 1-3)
g	inequality constraint function
h	equality constraint function
P	objective function for a subproblem (see eq. 4) or the system level objective (see eq. 5)
r	scaling factor used to implement equality constraints
T	set of order pairs of integers (see eqs. 1-3)
v	design variable
y	set of integers (see eqs. 1-3)

ρ tolerance factor (see eq. 4)

Introduction

The purpose of this paper is to describe the design and use of a computer program for multilevel optimization research. The work is motivated by the NASA commitment to large aerospace systems design. Multilevel optimization is a proposed method for reducing design time and improving design quality. The present multilevel simulator allows controlled experimentation with multilevel strategies.

Current projects in aerospace design are characterized by large numbers of design variables, complex mission requirements and computationally costly analyses. To solve these design problems on the current generation of computers requires a division of the task into subproblems. Traditionally, the problem is subdivided into disciplines then further subdivided into component analyses. Each discipline produces an optimized design then these preliminary designs are integrated into a final design. Multilevel optimization research aims to automate and formalize this accepted practice.

A number of multilevel decomposition and optimization strategies have been proposed. References 1-4 are representative papers which survey much of the available literature. Recently, several attempts at engineering design studies have used a multilevel approach[5,6]. It is clear that this method is promising and does produce results for some class of problems.

The purpose of the present simulator, version 1.0, is to evaluate different optimization strategies and classes of problems in a controlled

and efficient manner. As cited in reference 1, there are a number of aspects of multilevel optimization requiring research. The present simulator can investigate the overall convergence of these methods, the influence of coupling between the subsystems, the computational cost savings of multilevel compared to the single-level approach and input-output strategies. The simulator is also a guide for constructing new multilevel optimization codes. The FORTRAN computer program is modular and well documented. It demonstrates effective ways to implement multilevel optimization. Certain portions of the simulator code can be reused in other engineering applications.

This paper emphasizes the capabilities of the simulator computer program and the mechanics of preparing input and interpreting output. The theory of multilevel decomposition is explained only when necessary to define input and output quantities; a familiarity with the references is assumed.

Design of Multilevel Simulator

Problem Specification

This section describes the general class of problems handled by the simulator. The methods used to specify a multilevel decomposition and used to simulate engineering analyses are discussed. Many of the concepts discussed in this section are adapted from reference 1.

Figure 1 is a schematic of an optimization problem which is decomposed into three levels. There are seven design variables (v_i , i=1,7) and six

constraint functions($g_k \leq 0$, k=1,6). The problem is to minimize the system level objective, P_{11}, subject to the constraints. The single level approach would state the problem as:

$$\text{minimize} \quad P_{11}(v_1, v_2)$$

$$\text{STO} \quad g_1(v_1, v_2, v_3, v_4) \leq 0$$

$$g_2(v_1, v_2, v_3, v_4) \leq 0$$

$$g_3(v_1, v_2, v_5, v_6) \leq 0$$

$$g_4(v_1, v_2, v_5, v_6) \leq 0$$

$$g_5(v_3, v_4, v_7) \leq 0$$

$$g_6(v_3, v_4, v_7) \leq 0$$

Decomposition is possible whenever each constraint is a function of a proper subset of all design variables. Instead of a single optimization problem, this example is decomposed into four smaller optimization problems. Each subsystem is an unconstrained optimization problem which minimizes some cumulative measure of constraint violation. These cumulative constraint functions are denoted, P_{ij}, where i is the level number and j is the

subsystem number. The cumulative constraints are a function of the subsystem design variables and of fixed parameters.

The current computer program can simulate a multilevel optimization which has an arbitrary number of levels and an arbitrary number of subsystems on any level. The number of design variables and constraint functions tested is only limited by the users requirements for rapid processing. There are restrictions, however, on the decomposition methods. The simulator assumes that fixed parameters for cumulative constraint P_{ij} are design variables for at least one cumulative constraint in level i-1. For instance, in figure 1 the variable v_3 is a fixed parameter in subsystem 31 and a design variable in subsystem 21. Cross coupling is allowed (eg. v_3 can also appear in system 22) but reverse coupling is prohibited in the present version of the simulator (eg. v_7 cannot be a fixed parameter in any system above level 3). The interaction quantities defined in reference 1 are not included in from the present simulator.

The simulator uses explicit analytic functions of the design variables with analytic first derivatives to simulate constraint and objective functions. This is in contrast to realistic engineering applications in which the functions of the design variables are usually implicit and very expensive to evaluate. In this way the simulator can study the convergence behavior of very large multilevel optimization problems while using small amounts of computer resources.

The constraint functions can be either equality or inequality constraints and can have three different forms.

$$g_i = -a_i - \sum_{y_i} (b_i(k)v(k) - c_i(k)/v(k)) + \sum_{T_i} d_i(k,l)/v(k)/v(l) \quad (1)$$

$$g_i = -a_i + \sum_{y_i} (b_i(k)v(k)) + \sum_{T_i} d_i(k,l)v(k)v(l) \quad (2)$$

$$g_i = -a_i + \sum_{y_i} b_i(k)v(k)^{0.5} + \sum_{T_i} d_i(k,l)(v(k)\ v(l))^{0.5} \quad (3)$$

where a_i, b_i, c_i, and d_i are arrays of positive real numbers and y_i is a set of integers, k, which identify elements of the global design vector and T_i is a similar set of ordered pairs of integers, k and l. Summation over the set y_i implies that a different value of k is used for each term in the ith constraint function.

Equation 1 is a strictly decreasing function, equation 2 is strictly increasing and equation 3 increases to a positive-slope asymptote. Together, these three types of equations can simulate most engineering systems. The systems which cannot be simulated, such as those with disjoint domains, are difficult to optimize with any general purpose optimization code.

The cumulative constraints on the sublevels have the form of the Kreisselmeier-Steinhauser (K-S) function[7],

$$P_{ij} = \ln \left[\sum_k \exp(\rho g_k) \right] / \rho \quad (4)$$

where summation is over all inequality constraint functions in subsystem ij and over the cumulative constraints from the level below. The K-S tolerance factor, ρ, is a user-defined positive real number.

Equality constraints have the same forms as inequality constraints but must be handled differently. The notation $h_k = 0$ and $g_k \leq 0$ is used to separate the two types of constraint function. Equality constraints are implemented by adding a penalty function to the cumulative constraint and by adding an extra inequality constraint to the problem. Thus the unconstrained subsystem optimization problem:

$$\text{minimize} \quad P_{ij}$$

becomes a constrained problem:

$$\text{minimize} \quad P_{ij} \quad -[\, r \sum_{r} h_k \,]$$

$$\text{subject to} \quad h_k \leq 0$$

where r is a scaling factor chosen so that the components of the gradient of P_{ij} and the gradient of h_k have similar magnitudes. Notice that minimizing the objective function tends to increase the value of h_k while satisfying the constraint has the opposite effect. The net effect is to drive the value of h_k toward zero as required.

The objective functions are less flexible than the constraints. At the present the only system level objective function available is

$$P_{11} = \sum_{i} v(i)^2 \tag{5}$$

where the square of v(i) is summed over the set of all system level design variables. However, this objective function can be changed to any function

which has analytic first derivatives if the user is willing to make small modifications to the computer code.

Simulator Input and Output

In this section, simulator input and output is described. The number of levels, the number of subsystems on any level, the number and type of constraint functions are aspects of the test problem controlled by the user. The initial design vector and the side constraints on that design are easily modified as well. Finally, the amount and type of output can be selected. This section describes input and output decisions in general terms using the multilevel decomposition in figure 1 as an example. The appendix contains samples of the actual input and output files which correspond to the same test problem.

All of the information contained in figure 1 can be transferred to the simulator program interactively. The user answers questions about the total number of design variables (7) and the number of sublevels (2). For each level, beginning at the bottom, the user counts and lists all parameters that appear in that level. Then for each subsystem, the user defines the design variables and the number and type of constraint functions.

To totally specify the test problem represented by figure 1, the constraint functions must be associated with one of the three types of functions (eq. 1-3) and the matrices a,b,c,d and T must be defined. In order to limit the amount of input required of a user, the simulator uses a random number generator to fill the matrices. Only integer matrix T must be user defined. Notice that specifying entries in the T matrix is an easy and effective way to control the behavior of the constraint function under

changes in the design variable or parameter values. It is the user's responsibility to choose acceptable entries for each T matrix. The integers, k and l in equations 1-3, must be chosen from the list of design variables and fixed parameters for the current subproblem.

Once a test problem is designed, all matrices and function definitions are saved on a file and can be reused. The user can edit the file and change any of the coefficients in order to tailor the test problem to his needs. This is a very effective way to build a collection of test problems for multilevel research. With the test problem well defined, the effect of changes in optimization strategy or initial guess can be studied.

The simulator has a number of optional inputs which control the optimization process. The values are contained in a namelist called SIM on a data file called V.DAT. Table 1 lists the name, definition and default value for each input parameter. Most of the parameters have simple meanings. For instance, integers IPRINT and IPLOT control the amount of printed and graphical output respectively. The real arrays VORIG,VLB,and VUB specify an initial guess of design variables and an upper and lower bound on each. The meaning of other optional inputs will be clarified in the section on simulator implementation. Default values are usually sufficient for correct execution.

There are three kinds of simulator output: printed listing, graphical display and restart files. The graphical output is optional and all output is controlled by input parameters listed in table 1.

The printed listing is saved on a file P.LST which can be routed to a printer or reviewed on the screen. The listing contains the values of the

design variables at the start of each cycle and at the conclusion of the subsystem optimizations for that cycle. The value of the cumulative constraint for each subsystem, P_{ij}, and the number of iterations required to optimize the subsystem are also listed. Before the system level optimization occurs, the initial values of system level objective and design variables plus the maximum cumulative constraint from level 2 are listed and, optionally, plotted. If the process converges, then the final design vector and system level objective is printed. Otherwise, these values are listed during the execution of the following cycle.

The amount of computational work expended during the present cycle is tracked in two ways. First, the number of constraint evaluations is a count of the number of times any function g_i is evaluated starting at the beginning of a simulator run. Second, the elapsed time in seconds is the difference in the system clock time from the start to finish of the present cycle. Note that the number of function evaluations depends only on the input data, while the elapsed time depends on the status of the computer system as well. Thus if the same test case is run several times or on several different computers, the elapsed time is the only output that should change.

Graphical outputs appear on the screen at the beginning of the system level optimization for each cycle if the IPLOT parameter is set. At present, the plots are bar charts of system level design variables, system level objective and the maximum violated constraint value. By examining these charts as the simulator operates, the user can determine whether the

objective is being reduced and whether a feasible design is being reached. The plots also give some indication whether convergence is smooth or oscillatory.

The final type of output is a simulator restart file named OUT.DAT. This is simply a copy of the namelist input parameters with revised values in the VORIG array. The simulator can be restarted at the last cycle by merely renaming OUT.DAT to V.DAT.

Simulator Implementation

This section describes the simulator computer code.* Figure 2 contains a block diagram of simulator subroutines arranged by calling hierarchy and table 2 contains a list of routine names and purposes arranged alphabetically. The methods used in each routine will be summarized starting with the lowest level routines at the bottom of figure 2 and working to the top.

The constraint functions and their derivatives are analysed by routines CUM and GRAD. These routines evaluate eqs. 1-3 and their first derivatives analytically. The results are exact except for truncation errors caused by finite word size. Use of double precision arithmetic (REAL*8) is recommended to minimize this error. The constraint functions and derivatives are evaluated one by one and the cumulative constraint is calculated by routines OPTOBJ and OPTGRAD.

*The code was developed on a DEC MICROVAX II workstation in FORTRAN 77. The graphics were generated by GKS.

CONMIN[8] is a standard constrained optimization package widely used for scientific and engineering applications. It is used for both the subsystem and system level optimization. At the subsystem level, the conjugate direction method of Fletcher and Reeves is used to minimize the unconstrained cumulative constraint function. At the system level or at the subsystem level when equality constraints exist, CONMIN employs the method of feasible directions to solve the constrained optimization problem. All inputs to CONMIN are set to the values recommended in reference 8. The number of iterations allowed for any CONMIN optimization can be controlled by the namelist input parameters ITSYS and ITSUB. Limiting CONMIN to a few iterations may greatly reduce the effort needed to find a reasonable solution from a arbitrary initial design. More iterations may be necessary to refine that reasonable solution into the optimal point.

Subsystem processing is handled by OPTINP, OPT and SENSI. The first routine reads constraint function matrices from a file. The second calls CONMIN repeatedly until a minimizing solution or the maximum number of iterations is reached. The final routine analyzes the sensitivity derivatives for the solution. When there are no equality constraints, sensitivity derivatives are defined as the partial derivatives of the objective with respect to the fixed parameters. These partials are calculated by the routine DELP. When equality constraints exist, the method detailed in reference 9 is used. The routine SENSUB performs the needed operations and returns sensitivity derivatives.

The system level optimization routine, TOP, minimizes the system level objective subject to cumulative constraints from each level 2 subsystem. Unlike its subsystem level equivalent, OPT, this routine does not evaluate constraints analytically. As suggested in reference 1, the constraint values are estimated using linear extrapolations based on the change in system level variables and the value of sensitivity derivatives.

Move limits are added to the system level optimization because the linearly approximated constraint functions are valid only near the initial design. These move limits are implemented as CONMIN side constraints and can be adjusted using the the input parameter MOVFAC and the upper and lower bound arrays VUB and VLB. For each design variable, the move limit is computed by dividing the difference between the upper and lower bound by MOVFAC. No design variable is allowed to change by more than this amount.

In the present version of the simulator, the computed move limits are divided by the cycle number. In this way, the move limits can be ignored for the first cycle (eg. MOVFAC=1) and gradually tightened as a feasible solution is approached. Collapsing move limits forces the simulator to converge quickly to an improved design. If necessary, the simulator can be manually restarted until the global minumum is reached. The manual restart is accomplished by renaming the file OUT.DAT to V.DAT and executing the program again.

The value of the K-S tolerance factor, controlled by the namelist input parameter RHO, has a significant effect on the system level optimization. A small value of this factor (eg. RHO=1.0) means that the K-S function is a very smooth but very conservative envelope function of the constraints. As

RHO approaches infinity, the K-S function follows the constraint surface more closely and thus the design space enlarges but the slope of the function can change very abruptly. Under these latter conditions, the linear decomposition used in the system level optimization requires very tight move limits. Therefore, the values of input parameters MOVFAC and RHO should be increased or decreased together.

The main simulator routine, MULTI, controls the multilevel optimization process and controls input and output functions. The main routine invokes OPTINP, OPT and SENSI to perform subsystem level input and calculations. Occasionally, a call to OPT will be skipped. This occurs when the cumulative constraint from the level below is a positive number but is not a function of the current subsystem design variables. By skipping such a subsystem, needless function evaluations are bypassed and the unpredictable effect of a cumulative constraint (ie. the objective function) which has a zero gradient is avoided. The main routine calls for subsystem optimizations in order, beginning at the highest level. The main routine combines the objective function and sensitivity derivative information for all subsystems on the same level and passes that information on to the next higher level.

The main simulator routine also terminates the multilevel optimization process if convergence occurs or if the maximum number of cycles is exceeded. Convergence means that the largest cumulative constraint in level 2 (ie. P_{2j}) is less than some small positive number and the change in the system level design is negligible. In other words, the simulator stops

when a feasible design is identified by the sublevel optimizations and the system level optimization cannot improve on that design. Because of the way the move limits collapse in the system level optimization, this process may not converge to the best design. However, if convergence occurs, the solution is usually a feasible one which reduces the objective function. An improved design may be possible by restarting the simulator at this solution.

Results and Discussion

This section demonstrates simulator usage. The small multilevel optimization problem diagrammed in figure 1 is the sample problem. Input and output files for the sample problem are discussed. Printed output and plots of all cycles are included. The influence of the input parameter settings on the results and suggestions for modifying input parameter values is the final discussion topic.

Simulation Example

The following namelist input is contained in a file 'V.DAT':

```
$SIM
NLEVELS=2, LENV=7, IGEN=0, IFILE='TMX.DAT', IPLOT=1, IPRINT=0,
ITSYS=10, ITSUB=10, NCYCLE=10, MOVFAC=6, RHO=3.0,
VORIG=7*1.0,  VLB=7*0.1,  VUB=7*2.0
$END
```

This file specifies a simulator model using namelist variables defined in table 1. The model has 2 sublevels below the system level and has a total of seven design variables. The model already exists and is found in a file named 'TMX.DAT'. The interactive inputs used to create this file are reproduced in the appendix. Plotted output and the normal simulator output

are selected but all CONMIN output is suppressed. Both system level and subsystem level optimization processes are limited to 10 iterations and the entire multilevel optimization is limited to 10 cycles.

Each of the seven design variables is initially set to unity and is allowed to vary from a lower bound of 0.1 to an upper bound of 2.0. At the system level, the design variables are limited to one sixth of this range according to the value of MOVFAC. The K-S tolerance factor is set to a moderate value of 3.

The printed output file 'P.LST' is reproduced and annotated in the appendix and discussed here. For each cycle, the original design is given followed by a summary of results of subsystem optimizations. For instance, the first subsystem in level 2 stopped after five CONMIN iterations with a value of 0.3926 for the objective function. This objective function is the cumulative constraint, P_{21}, for all the constraint functions in the subsystem plus the linear approximation to the cumulative constraint, P_{31}, from level 3. Notice that the second subsystem of level 2 has an objective function value of 0.0 after 0 iterations. This means that this subsystem was skipped. Also included in the summary of subsystem results is the final value of design vector, v, and the maximum cumulative constraint from level 2. For the first cycle, the maximum constraint was 2.2114 which indicates that the design is not feasible.

Following the subsystem level results, the system level objective and the number of system level CONMIN iterations are reported. For instance, in the first cycle the objective has a starting value of 2.0. A new design is

reached after 10 interations. The whole cycle required 2.00 seconds and 68 evaluations of the constraint functions, g_k.

Cycle 2 can be interpreted in a similar manner. The process converges after cycle 2. Notice that the maximum cumulative constraint of -.0045 indicates a feasible design is produced by the sublevel iterations. When the top level optimization fails to find a new design vector which will significantly reduces the objective, the process terminates.

Figure 3 is an example of the plotted output produced by the simulator. The bar chart represents system level objective, design variable values and constraint values. The bar marked OBJ is the initial system level objective. The bars labelled DV1 and DV2 indicate the values of the two system level design variables. The bar labelled GMAX is the maximum cumulative constraint from the second level subproblems. A dashed line indicates that GMAX has a negative value. Notice that the plots are produced before the top level optimization begins. In this way, GMAX is an actual computed value for the maximum constraint which corresponds to the initial values of the system level variables. The alternative is to plot the optimized system level design with an extrapolated value of GMAX. The present method is thought to be more informative and is certainly more exact. The drawback is that the final design computed by cycle 2 is not plotted automatically. Restarting the simulator from the final solution will produce this plot.

Figures 3(a-c) contain the plotted output which corresponds to the printed output discussed above. In both outputs, it is clear that the

initial design is infeasible and has an objective of 2.0. The next design is feasible and the system level objective is reduced to 1.0. At the end of two cycles, the process converges to another feasible design with an objective of .8898. By restarting the simulator with larger values for RHO and MOVFAC it can be discovered that .8898 is not the best system level objective possible. Figure 3d illustrates the actual solution to this problem.

Figure 4 is an alternate way to interpret simulator results. This figure indicates how the system level objective function and constraint functions vary with system level design variables. To produce the figure, the design variables v_3, v_4, v_5, v_6 and v_7 are fixed at the final values computed by the simulator. The system level design variables are varied over the allowable range from 0.1 to 2.0. A dashed line is used to represent the system level objective function and solid curves are used for constraint functions. Only the active and violated constraint functions appear in the plot.

Figure 4 corresponds to the design shown in figure 3c. Figure 4a shows the change in function values with the first design variable, v_1. Figure 4b is a similar plot with v_2 as the ordinate. On each plot, the asterisk on the objective function curve denotes the solution point. Notice that the solution point is relatively far from any constraint curve. This is because of the small value of the K-S function tolerance parameter (ie. RHO=3.0). If the K-S function were plotted instead of the actual constraint functions,

then the solution point would seem more sensible. Clearly, the simulator needs to be restarted with a larger value of RHO.

Control of Simulator Operation

Several types of convergence behavior occur during simulator use depending on the test case, on the initial design vector and on the values of input control variables (see Table 1). Starting from an arbitrary design point, a reasonable value for RHO is about 3.0 and a reasonable value for MOVFAC is about twice the value of the largest entry in the VUB array. If the simulator fails to converge and if the plotted value of GMAX is not improving, then the move limits are probably too loose. Restarting the simulator with a larger value of MOVFAC is advised. On the other hand, if the simulator finds a feasible solution after one or two cycles, then there are two possible explanations. Either, the initial design is very close to the optimal solution by chance, or the move limits are so restrictive that the actual solution is out of reach. Restarting from the new solution without any change to RHO or MOVFAC will clarify the situation. If a near optimal point has been reached, then the simulator can be restarted with gradually increased values of RHO and MOVFAC which will refine the solution. If, however, the objective can be reduced significantly by restarting with low values of RHO and MOVFAC, then this should be continued until no more change occurs. At this point, RHO and MOVFAC can be increased as suggested above.

More serious difficulties with the simulator are possible. First, since the test problem is created using a random number generator, the problem may not have a feasible solution. This possiblity should be

suspected if at least one sublevel constraint is positive for all cycles and for many different initial designs. A second bothersome condition is an overflow error which occurs in the CONMIN subroutine. This condition is related to the use of linear approximations in the system level optimization. Restarting with a smaller value of RHO and the current value of MOVFAC is usually helpful. A final difficulty will occur if the test problem is incorrectly generated. For instance, the user might have included elements in the T matrix which were not listed as either fixed parameters or as variables in that subproblem. The results of all such input errors can not be predicted.

Concluding Remarks

Designing the large aerospace systems of the future will require optimization techniques which are at least an order of magnitude more capable than those presently available. Multilevel optimization techniques promise to address these needs, however, much research remains. The present version of the multilevel simulator can generate test problems which have the characteristics of large engineering optimization problems but which require a few seconds of computer processing time rather than hours or even days. In this way, the performance of multilevel optimization techniques can be studied and improved for a wide variety of problems.

The simulator computer program, described in this document, can generate test problems having an arbitrary number of levels and an arbitrary number of subsystems on each level. The number of design variables and constraint functions is unrestricted but the decomposition of these

constraints into subproblems is not particularly flexible in the current version of the simulator. It is anticipated that future users of the simulator will modify the code in order to add new system level objective functions or, to test new optimization packages, or apply new convergence criteria or to incorporate other advanced multilevel techniques. For this reason, and because the code is useful as a framework for new multilevel applications, the code is modularized and documented.

Appendix

This appendix contains sample input and output files which correspond to the test problem described in the paper. Figure 1 is a schematic of the test problem.

The first step in simulating the test problem is to prepare the following namelist input file, named V.DAT.

```
$SIM
NLEVELS=2, LENV=7, IGEN=1, IFILE='TMX.DAT', IPLOT=1, IPRINT=0,
ITSYS=10, ITSUB=10, NCYCLE=10, MOVFAC=6, RHO=3.0,
VORIG=7*1.0,  VLB=7*0.1,  VUB=7*2.0
$END
```

Since input parameter IGEN is set to 1 (see Table 1), the simulator will collect additional information from the user and build a new test problem description file named 'TMX.DAT'. A portion of the interactive exchange and the resulting description file are reproduced below. Bold face type is used to distinguish user responses from simulator generated text.

```
 USER INTEGER RESPONSES SEPARATED BY COMMAS


 THIS IS LEVEL           3
 ENTER THE NO. OF SUBSYSTEMS ON THIS LEVEL
 1
ENTER TOTAL NO. OF PARAMETERS THIS LEVEL
 2
 WHICH ELEMENTS OF THE V VECTOR ARE THEY?
 3,4
THIS IS SUBSYSTEM           1
 ENTER NO. OF VARIABLES
 1
 WHICH ELEMENTS ARE THEY?
 7
ENTER NUMBER OF CONSTRAINT FUNCTIONS
 2
 CONSTRAINT FUNCTION           1
  IS CONSTRAINT (1-DECREAS,2-INCREAS,3-ASYMP) ?
 1
ENTER NO. OF PAIRS IN THIS T MATRIX
 4
ENTER NEXT PAIR FOR T MATRIX
 7,7
ENTER NEXT PAIR FOR T MATRIX
 7,3
ENTER NEXT PAIR FOR T MATRIX
 3,7
ENTER NEXT PAIR FOR T MATRIX
 4,3
 CONSTRAINT FUNCTION           2
  IS CONSTRAINT (1-DECREAS,2-INCREAS,3-ASYMP) ?
 2
ENTER NO. OF PAIRS IN THIS T MATRIX
 3
ENTER NEXT PAIR FOR T MATRIX
 7,7
ENTER NEXT PAIR FOR T MATRIX
 3,4
ENTER NEXT PAIR FOR T MATRIX
 7,4


 THIS IS LEVEL           2
 ENTER THE NO. OF SUBSYSTEMS ON THIS LEVEL
 2
ENTER TOTAL NO. OF PARAMETERS THIS LEVEL
 2
 WHICH ELEMENTS OF THE V VECTOR ARE THEY?
```

```
 1,2
THIS IS SUBSYSTEM          1
 ENTER NO. OF VARIABLES
 2
       etc.   ...
```

A portion of the file, TMX.DAT, which was generated by the interactive responses above is included next:

```
    1
    2
         3          4
    1
         7
    2
    4    1
    7    7
    7    3
    3    7
    4    3
    3    2
    7    7
    3    4
    7    4
  0.27226
  0.42816   0.31650   0.87439   0.31424   0.17683   1.09099   1.02129
          etc.   ...
```

The solution to the test problem and its interpretation are discussed in the body of this paper. A complete listing of the output file is included here. Notice that the input file V.DAT is printed at the beginning of this listing. The value of parameter IGEN is now set to 0 because the input file TMX.DAT already exists. Annotations are added in the margin to guide the reader. Figure 3 contains a graphical presentation of the same results.

```
$SIM
IPRINT  =          0,
ISENS   =          0,
ITSYS   =         10,
ITSUB   =         10,
NCYCLE  =         10,
NLEVELS =          2,
LENV    =          7,
RHO     =   3.000000    ,
VORIG   = 7*1.000000    , 20*0.0000000E+00,
VLB     = 7*0.1000000   , 20*0.0000000E+00,
VUB     = 7*2.000000    , 20*0.0000000E+00,
IPLOT   =          1,
MOVFAC  =          6,
IGEN    =          0,
IFILE   = 'TMX.DAT   '
$END
```

> Input file

```
V=    1.0000   1.0000   1.0000   1.0000   1.0000   1.0000   1.0000
BEGIN CYCLE    1
------------------------

LEVEL=  3 subsystem= 1 OBJ including PENALTY =     0.3365  ITER=  4
LEVEL=  2 subsystem= 1 OBJ including PENALTY =     0.3926  ITER=  5
LEVEL=  2 subsystem= 2 OBJ including PENALTY =     0.      ITER=  0

V=    1.0000   1.0000   1.3167   0.6833   1.0000   1.0000   0.9817

SYSTEM LEVEL OBJECTIVE=    2.0000
MAX. CUM. CONSTRAINT FROM LEVEL 2 =    0.22114E+01
DESIGN IS FEASIBLE IF IT IS LESS THAN    0.10000E+00

SYSTEM LEVEL ITERATIONS= 10
ELAPSED TIME IN SECONDS FOR THIS CYCLE =              2.00000
TOTAL NO. OF CONSTRAINT EQ. EVALUATIONS=                  68

        ************************************************************

V=    0.6833   0.7307   1.3167   0.6833   1.0000   1.0000   0.9817
BEGIN CYCLE    2
------------------------

LEVEL=  3 subsystem= 1 OBJ including PENALTY =     0.0802  ITER=  4
LEVEL=  2 subsystem= 1 OBJ including PENALTY =    -0.0730  ITER=  5
LEVEL=  2 subsystem= 2 OBJ including PENALTY =     0.1631  ITER= 10

V=    0.6833   0.7307   1.6333   0.3667   1.1631   0.2998   0.9767

SYSTEM LEVEL OBJECTIVE=    1.0008
MAX. CUM. CONSTRAINT FROM LEVEL 2 =   -0.45224E-02
DESIGN IS FEASIBLE IF IT IS LESS THAN    0.10000E+00

SYSTEM LEVEL ITERATIONS=  4
ELAPSED TIME IN SECONDS FOR THIS CYCLE =              0.85938



TOTAL NO. OF CONSTRAINT EQ. EVALUATIONS=                 212

        ************************************************************

PROCESS CONVERGES AFTER CYCLE  2
FINAL SYSTEM LEVEL OBJECTIVE=    0.8898
FINAL DESIGN VECTOR
V=    0.5250   0.7837   1.6333   0.3667   1.1631   0.2998   0.9767
LEVEL=  3 subsystem= 1 OBJ without PENALTY =    -0.0974 GMAX =    -0.3251
LEVEL=  2 subsystem= 1 OBJ without PENALTY =    -0.5773 GMAX =    -0.5773
LEVEL=  2 subsystem= 2 OBJ without PENALTY =     0.0325 GMAX =    -0.0969
```

> see Fig 3a

> see Fig 3b

> see Fig 3c

References

1. Sobieszczanski-Sobieski, J.: A Linear Decomposition Method for Large Optimization Problems--Blueprint for Development. NASA TM 83248, Feb. 1982.

2. Kirsch, U.; Reiss, M.; and Shamir, U.: Optimum Design by Partitioning into Substructures. J. of Str. Div. ASCE, 1972, p. 249.

3. Schmidt, L. A.; and Ramanathan, R. K.: Multilevel Approach to Minimum Weight Design Including Buckling Constraints. AIAA J., Vol. 16, No. 2, Feb. 1973, pp. 97-104.

4. Haftka, R.T.: An Improved Computational Approach for Multilevel Optimum Design. J. Struc. Mech., vol. 12, no.2, pp. 245-261, 1984.

5. Sobieszczanski-Sobieski, J.; James, B. B.; and Riley, M. F.: Structural Optimization by Generalized Multilevel Optimization. Proceedings of the 26th AIAA/ASME/ASCE,AHS Structures, Structural Dynamics and Materials Conference, Vol. 1, pp. 349-365, April 1985.

6. Barthelemy, J.-F. M.; and Riley, M. F.: An Improved Multilevel Optimization Approach for the Design of Complex Engineering Systems, AIAA Paper 86-0950-CP, presented at the AIAA/ASME/ASCE/AHS 27th Structures, Structural Dynamics and Materials Conference, San Antonio, Texas, May 19-21, 1986.

7. Kreisselmeier, G.; and Steinhauser, R.: Systematic Control Design by Optimizing a Vector Performance Index. Proceedings of IFAC Symposium on Computer Aided Design of Control Systems, Zurick, Switzerland, 1979, pp. 113-117.

8. Vanderplaats, G. N.: CONMIN-- A FORTRAN Program for Constrained Function Minimization: User's Manual. NASA TMX-62282, Aug. 1973.

9. Barthelemy, J.-F. M.; and Sobieszczanski-Sobieski, J.: Optimum Sensitivity Derivatives of Objective Functions In Nonlinear Programming. AIAA Journal, Vol. 21, No. 6, pp. 913-915.

NAME	MEANING	DEFAULT
IFILE	NAME OF INPUT OR OUTPUT FILE (C) MEANING DEPENDS ON IGEN FLAG	'IN.DAT'
IGEN	TEST PROBLEM GENERATION FLAG (I) (0=INPUT FILE EXITS, 1=GENERATE FILE)	0
IPRINT	PRINT FLAG FOR CONMIN (I) (0=NO CONMIN OUTPUT)	0
IPLOT	PLOT FLAG (I) (0=NO PLOT)	0
ISENS	SENSITIVITY ANALYSIS SELECTOR (I) (0=SUPPRESS ANALYSIS, 1=ALLOW ANALYSIS)	0
ITSUB	MAX NO OF CONMIN ITERATIONS AT SUBSYSTEM LEVEL (I)	10
ITSYS	MAX NO OF CONMIN ITERATIONS AT SYSTEM LEVEL (I)	10
LENV	NO OF ELEMENTS IN DESIGN VECTOR (I)	0
MOVFAC	MOVE LIMIT ADJUSTMENT FACTOR (I)	1
NCYCLE	MAX NO OF MULTILEVEL CYCLES (I)	10
NLEVELS	NO OF SUBLEVELS (I)	0
RHO	TOLERANCE PARAMETER IN THE K-S FUNCTION (R)	1.0
VLB	LOWER BOUND FOR EACH ELEMENT IN DESIGN VECTOR (R)	-
VORIG	INITIAL GUESS FOR DESIGN VECTOR (R)	-
VUB	UPPER BOUND FOR EACH ELEMENT IN DESIGN VECTOR (R)	-

Note: (I)= INTEGER and (R)= REAL and (C)= CHARACTER STRING

Table 1. Simulator control parameters

Name	Purpose
CUM	Evaluates constraint functions
DELP	Calculates partial derivatives of cumulative constraint with respect to fixed parameters
FILLER	Uses random number generator to fill coefficient matrices
GRAD	Calculates the gradient of a constraint function
MODEL	Builds a new test case for simulator
MULTI	Supervises multilevel optimization
OPT	Supervises subsystem level optimization
OPTGRAD	Calculates partial derivatives of cumulative constraint with respect to subsystem variables
OPTINP	Reads input for a subsystem from a file
OPTOBJ	Calculates the cumulative constraint for a subsystem
SENSI	Calculates sensitivity derivatives
SENSUB	Performs sensitivity analysis (see ref. 6)
SETDEF	Sets CONMIN inputs to default values (see ref. 5)
TOP	Supervises system level optimization

Table 2. List of Simulator Subroutines

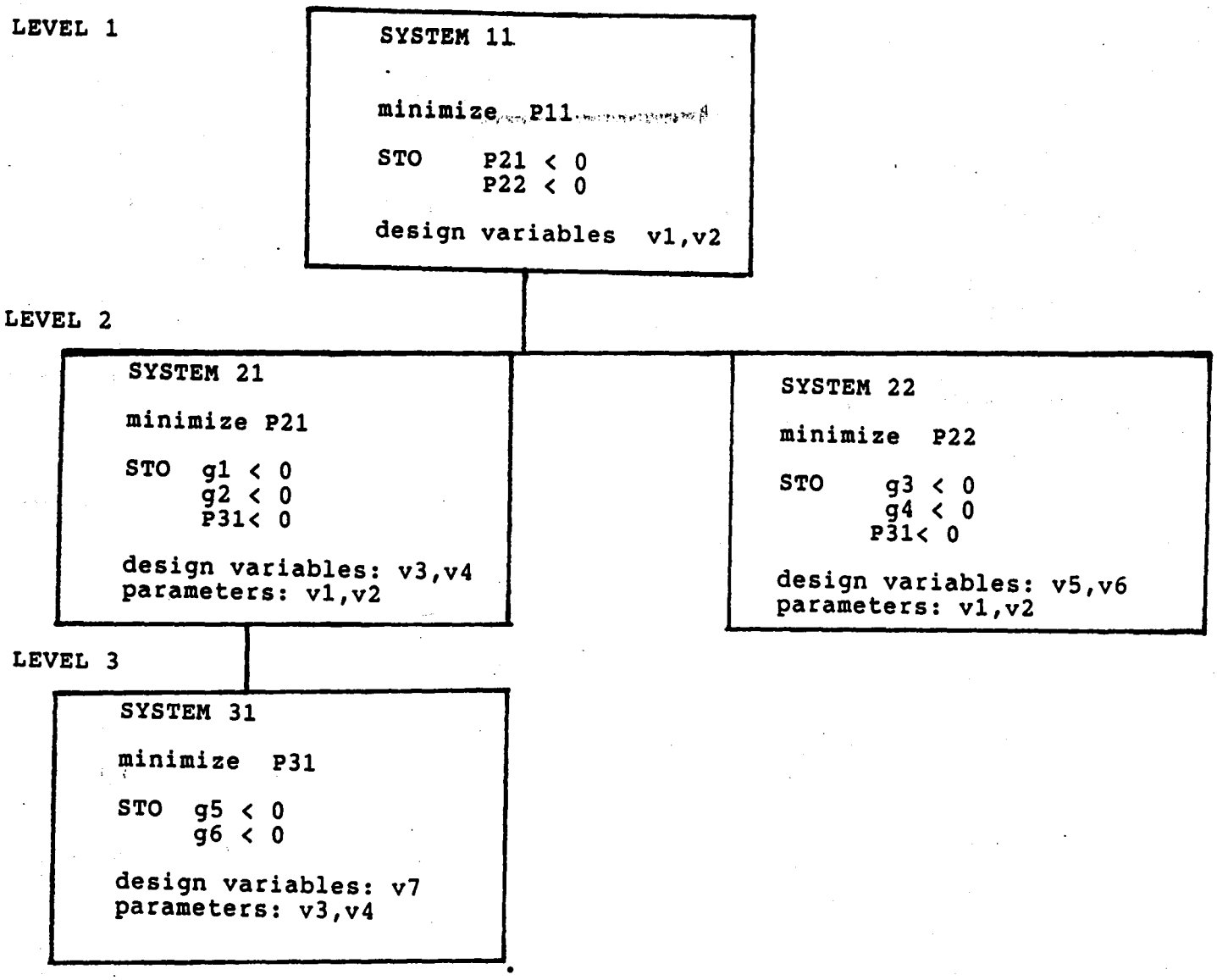


Figure 1. Schematic of Multilevel Problem

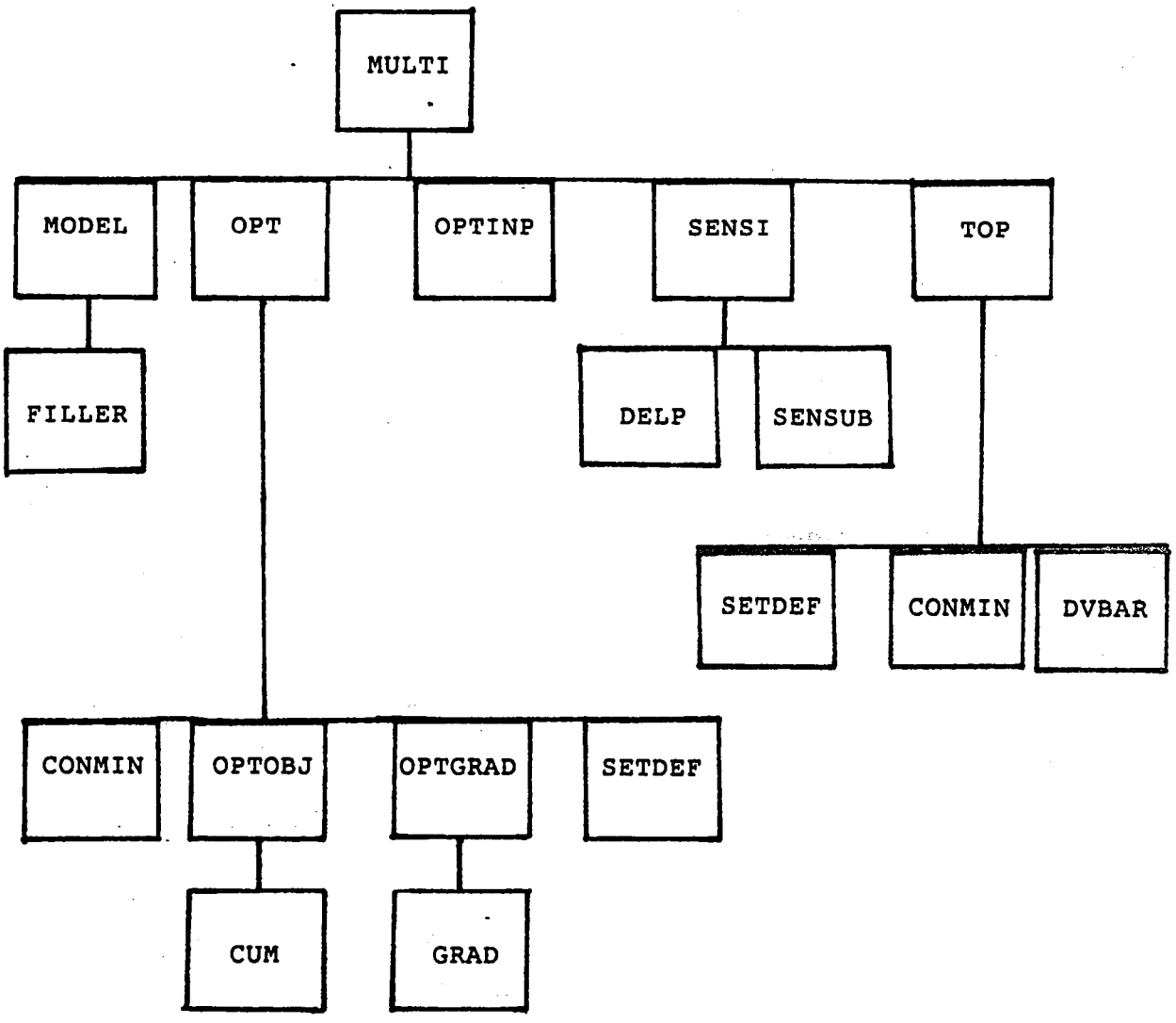
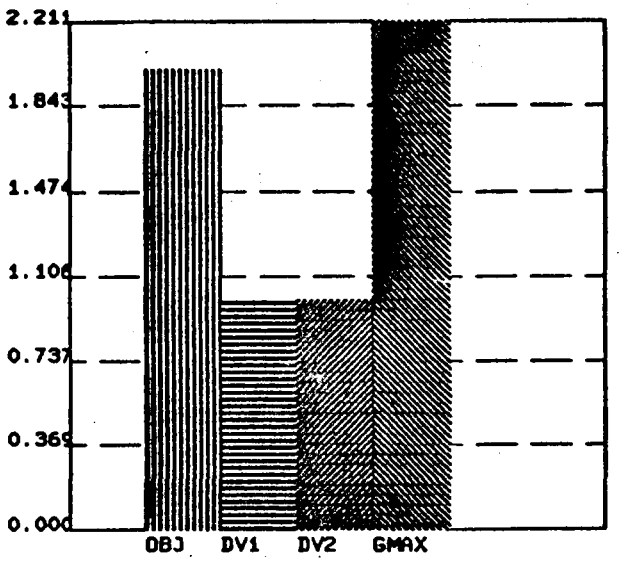
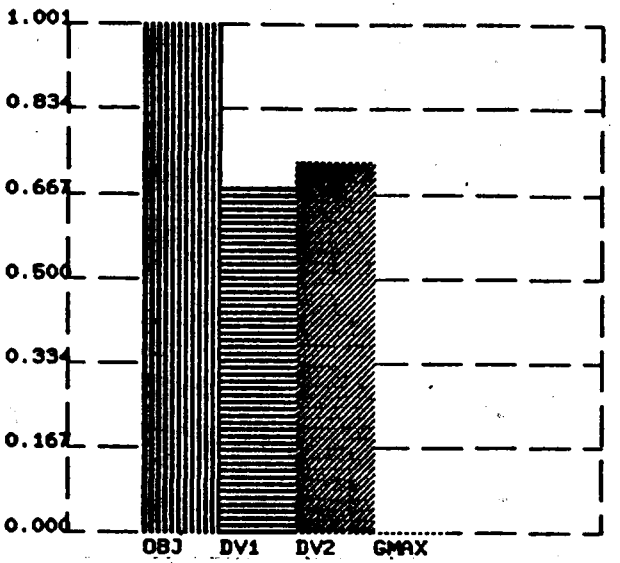


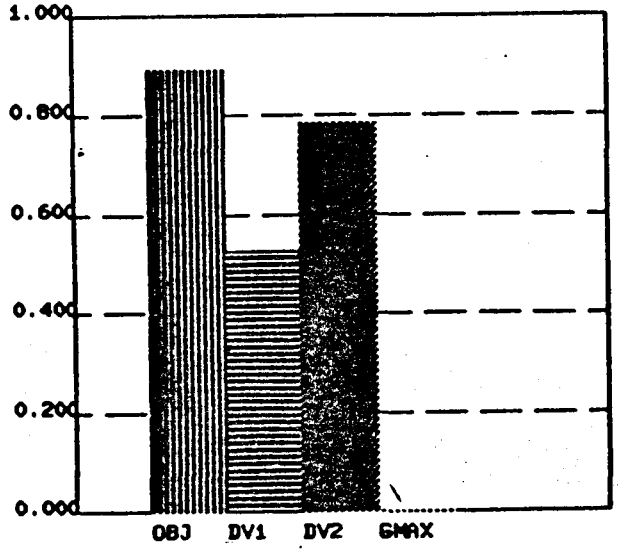
Figure 2. Simulator Subroutine Hierarchy



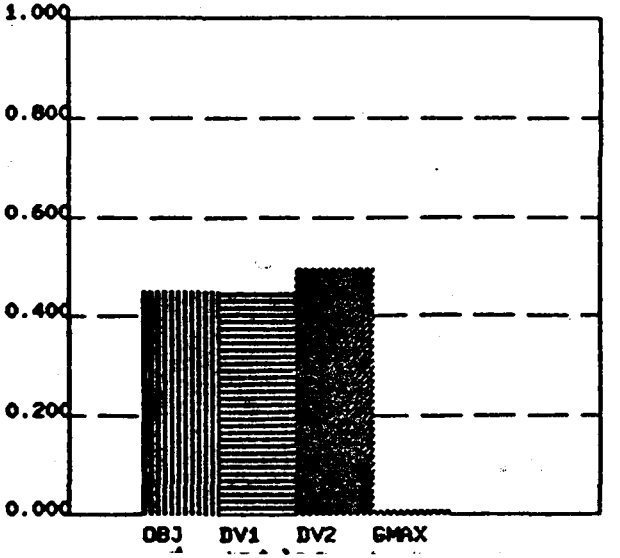
(a) Initial design



(b) Design after one full cycle

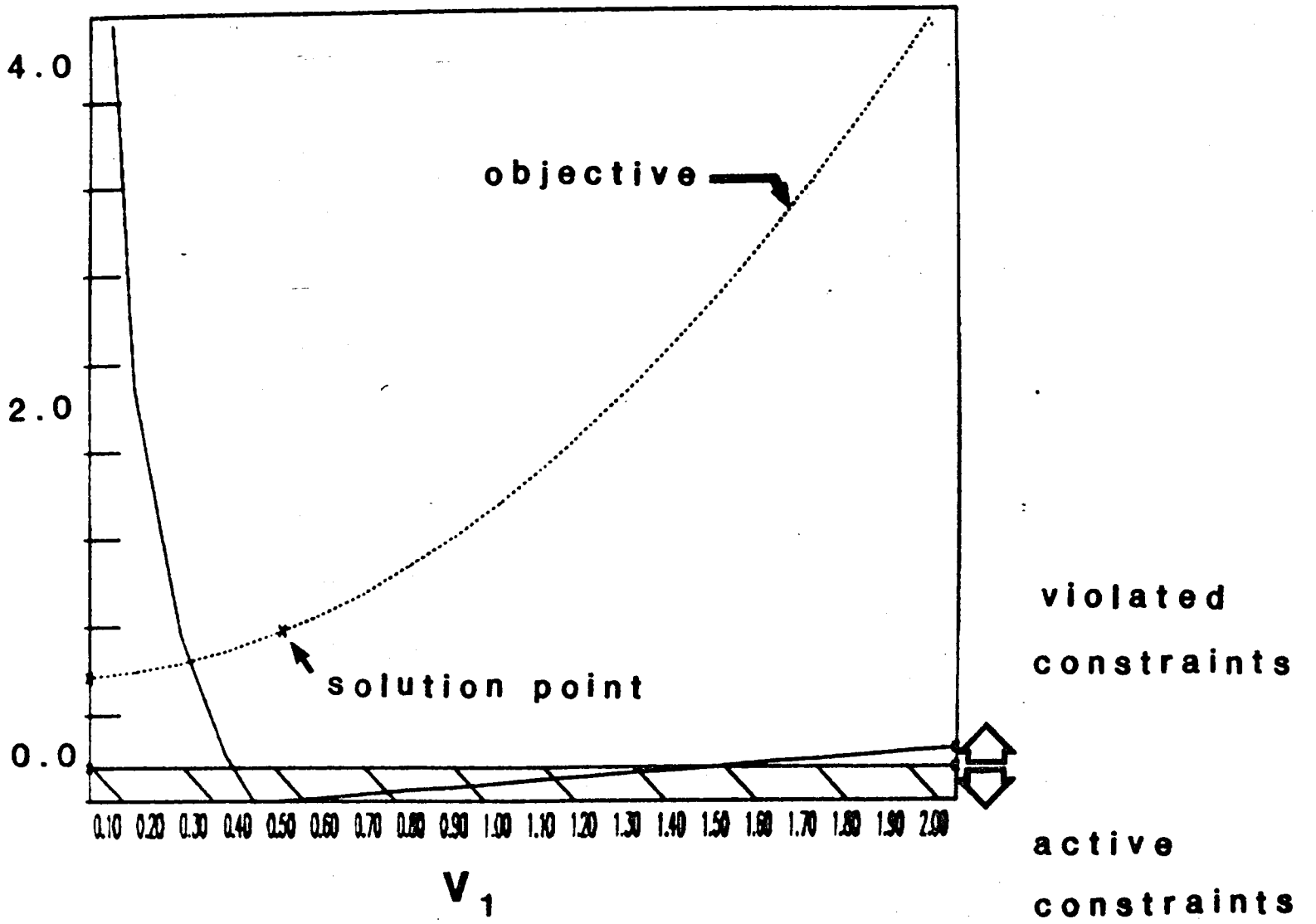


(c) Design after two full cycles



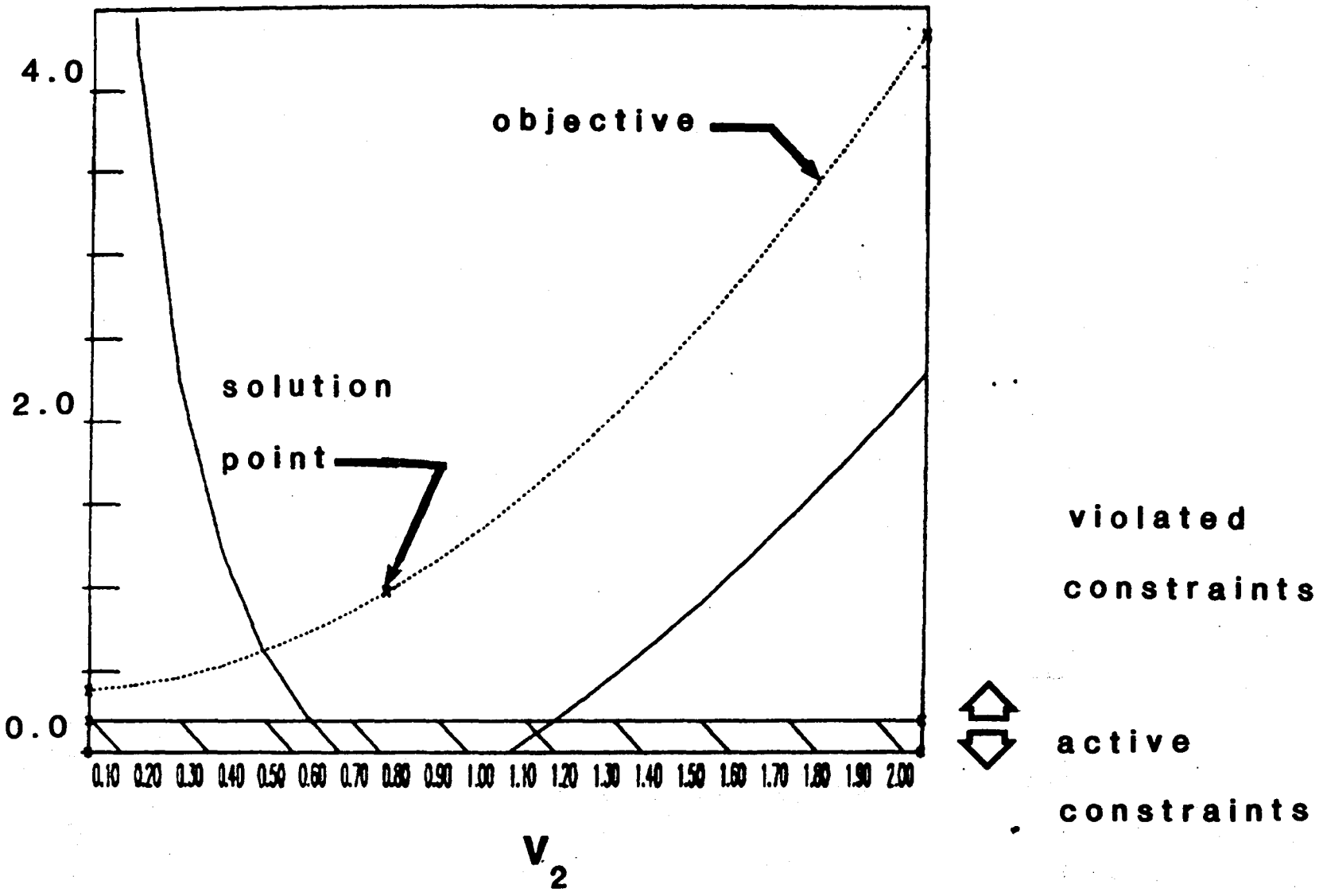
(d) Optimal design

Figure 3. Bar charts of system level results



(a) Change in functions with the first system level variable

Figure 4. Objective and constraint functions after two cycles



(b) Change in functions with the second system level variable

Figure 4. Concluded.

1 Report No.	2. Government Accession No.	3. Recipient's Catalog No.
NASA TM-87751		

4 Title and Subtitle	5. Report Date
Simulator for Multilevel Optimization Research	June 1986
	6. Performing Organization Code 505-63-11-01

7. Author(s)	8. Performing Organization Report No.
S. L. Padula and K. C. Young	
	10. Work Unit No.

9. Performing Organization Name and Address	11. Contract or Grant No.
NASA Langley Research Center Hampton, VA 23665	
	13. Type of Report and Period Covered

12. Sponsoring Agency Name and Address	13. Type of Report and Period Covered
National Aeronautics & Space Administration Washington, DC 20546	Technical Memorandum
	14. Sponsoring Agency Code

15. Supplementary Notes

16. Abstract

A computer program designed to simulate and improve multilevel optimization techniques is described. By using simple analytic functions to represent complex engineering analyses, the simulator can generate and test a large variety of multilevel decomposition strategies in a relatively short time. This type of research is an essential step toward routine optimization of large aerospace systems.

The paper discusses the types of optimization problems handled by the simulator and gives input and output listings and plots for a sample problem. It also describes multilevel implementation techniques which have value beyond the present computer program. Thus, this document serves as a users' manual for the simulator and as a guide for building future multilevel optimization applications.

17. Key Words (Suggested by Author(s))	18. Distribution Statement
Optimization, computerized simulation Operations Research, mathematical programing Computer program, FORTRAN	Unclassified - Unlimited Star Category 61

19. Security Classif. (of this report)	20. Security Classif. (of this page)	21. No. of Pages	22. Price
Unclassified	Unclassified	34	A03

N-305

For sale by the National Technical Information Service, Springfield, Virginia 22161

End of Document