Int. J. Solids Structures Vol. 26, No. 9/10, pp. 1173–1210, 1990
Printed in Great Britain.

0020–7683/90 $3.00+ .00
Pergamon Press plc

GENOPT—A PROGRAM THAT WRITES USER-FRIENDLY OPTIMIZATION CODE

DAVID BUSHNELL
Department 93-30/251, Lockheed Palo Alto Research, 3251 Hanover Street, Palo Alto, CA 94304, U.S.A.

Abstract—The purpose of GENOPT (GENeral OPTimization) is to enable an engineer to create a user-friendly system of programs for analyzing and/or optimizing anything. The application of GENOPT is not limited to the field of structural mechanics. GENOPT is designed to handle problems with small data bases, not large finite element models, although it might well be used to provide a user-friendly "shell" within which any analysis could be done. GENOPT is ideal for generating programs for optimizing objects the behavior (stress, buckling, vibration, etc.) of which can be expressed by relatively simple tables or formulas such as those that appear in handbooks, or for optimizing objects the behavior of which has been previously encoded in existing subroutines. The optimizer used in GENOPT, created by Vanderplaats, is called ADS. Two examples are presented, one relatively simple, the other more complex.

INTRODUCTION

When engineers embark on the task of designing an object that must survive certain environments during service, they often develop or use computer programs that analyze given, fixed configurations. If, under the various applied loads, a configuration appears to be inadequate, certain dimensions, materials, or other parameters are changed by the engineer and the analysis is repeated. The design evolves by means of this "manual" iterative process, which may require the expenditure of much labor and the passage of many days. Should a certain configuration prove satisfactory, that is, should the analysis program or programs show that this configuration survives all the environments with adequate margins of safety, the engineer is sorely tempted to terminate the iteration process and accept the design even though, while feasible, it may not be optimum.

The step from simple analysis to automated optimization seems to be a difficult one for many engineers and their managers. In every field there are many proven computer programs that analyze things with given configurations and given environments. Therefore, workers in a field are willing to accept the results of computerized analyses and to use these results as a basis for decisions on what to do next. However, these engineers and managers often seem less willing at present to allow the computer to make decisions about how the dimensions and other properties of the configuration are to be changed in order to minimize cost, minimize weight, or meet some other objective. Entering the field of automated optimization seems scary and appears to require mathematical expertise not available in time to meet the goals of a design project.

The main purpose of GENOPT is to make this step into the world of automated optimization easy. If the engineer has formulas from handbooks such as Roark's (1954), design curves, and/or algorithms for predicting the behavior of given configurations, GENOPT, working with these analysis tools, will generate a program system that can find the "best" design in a user-friendly way.

The user of GENOPT does not need to know much about optimization as a mathematical discipline. The optimizer used in GENOPT is called ADS. It was created by Vanderplaats (Vanderplaats and Sugimoto, 1986; Vanderplaats, 1987). ADS is "hardwired" in the "0–5–7" mode, which is the reliable "modified-method-of-feasible-directions" branch of this widely used optimization software.

The user of GENOPT must establish:

(1) what types of behavior should be included in the analysis?
(2) what allowable limits should be assigned to each type of behavior?
(3) what factor of safety should be assigned to each type of behavior?
(4) what is the objective of the optimization?

Over the past several years the author has developed several program systems written in FORTRAN that allow the user to analyze and optimize certain kinds of structures (Bushnell, 1983a, b, 1986a, b; 1987, 1988). These program systems have a similar style and structure. In a preprocessor, usually called "BEGIN", the user is asked in an interactive mode to supply a starting design, material properties, boundary conditions and loads. In a second processor called "DECIDE" the user is asked in an interactive mode to choose decision variables, lower and upper bounds of decision variables, linked variables, and something called "escape" variables (variables that are iteratively increased in size by 10% per iteration until optimization via ADS takes over). Then in a third processor called "MAINSETUP" the user is asked in an interactive mode to provide certain strategy parameters for the batch-mode analysis to follow. Finally, the user launches a batch run involving a mainprocessor. If the problem is an optimization problem, this batch run will yield a new design. Three of the program systems (Bushnell, 1983a, b, 1987, 1988) have a "CHANGE" processor which permits the user to assign new values to certain of the variables without having to run a case from "BEGIN". The three program systems (Bushnell, 1983a, b, 1987, 1988; have in common a user-friendly feature that definitions of the data appear throughout the output so that this output is easy to understand.

The idea is close at hand to offer the engineer the potential for developing analysis/optimization program systems that have a similar user-friendly style and structure. It should be possible to write a "master" computer program that, eliciting certain information from its user, generates program systems that perform tasks analogous to those performed by the program systems described in Bushnell (1983a, b, 1987, 1988) tasks that are defined by the user of this "master" program. This paper describes the implementation of this idea. Further details appear in files called GENOPTST.ORY, GENOPT.HLP, PLATE.*, PLATE 1.*, PANEL.*, ARIANE.* and HOWTO.RUN included with other files in the GENOPT program system (Bushnell, 1989).

THE ROLES OF USERS

There are two types of user that are involved here, called "GENOPT user" and "end user". The GENOPT user conceives of a generic class of objects for which a user-friendly ("end user" -friendly) analysis and/or optimization code is needed. He or she decides what types of behavior must be accounted for, from where program coding should be obtained to predict these types of behavior, what input data are needed, what name, definition, and "help" paragraphs should be associated with each input datum, and how the input data should be organized. If the GENOPT user performs these tasks well, GENOPT generates a user-friendly program system that will yield well documented, reliable analyses and/or optimum designs. The end user finds optimum designs of specific items that fit within the generic class conceived by the GENOPT user. The next two subsections provide more details on the roles of each of these two types of user. It is expected that the end user will often be the same person as the GENOPT user. Both type of user are GENOPT users in a literal sense: they both use GENOPT. However, they use GENOPT in different ways, as listed in Table 1.

Role of the GENOPT user

The GENOPT user designs the framework for optimization. He or she must perform the following tasks:

(1) Choose a generic class of problems for which an analysis and/or optimization program is needed. This class should not be so narrow as to be practically useless, nor should it be so broad that the end product is difficult to use, or overly expensive to run.

(2) Decide which phenomena (behaviors) may affect the design. Since these behaviors may constrain the design they are commonly termed "behavioral constraints" or just "constraints". For example, a certain structure may experience (a) instability, (b) material failure, (c) unacceptable modal vibration, (d) unacceptably large displacement, (e) high drag, (f) violation of clearance tolerances, etc. It is very important that every conceivable

Table. 1. A typical GENOPT (GENeral OPTimization) runstream. Commands typed by the user are preceded by "$"

```
================================================================
   COMMAND              PURPOSE OF THE COMMAND
----------------------------------------------------------------
$ GENOPTLOG    (Activate the GENOPT commands.)

SECTION 1      (TASKS TO BE PERFORMED BY THE GENOPT USER.)

First, decide upon a class of things for which there is a need for a
user-friendly analysis/optimization program.  Then consider the
following:

(a) What variables (length, width, thickness,...) are involved?
(b) What environments (loads, temperatures,...) is the class of
    things subjected to?
(c) What types of behavior (stress, buckling, vibration,...)
    should be accounted for?
(d) What is the objective (weight, cost,...) of the optimization?
(e) From where are you going to obtain algorithms for predicting
    the types of behavior in (c) and the objective in (d)?
    In brief, you must have a good idea of your goal before you start.
    You don't need quantitative answers, only a well-formulated plan.

$ GENTEXT

During an interactive session, provide data names, definitions,
and help paragraphs for a problem class for which you establish a
generic name. The GENOPT system sets up a data base, a system of
FORTRAN programs, and a prompting file, all of which use data names,
definitions, and ''help'' paragraphs created by you. You are asked by
GENTEXT to provide data names for variables each of which performs
one of seven possible roles:

1 = a possible decision variable for optimization, typically a
    dimension of a structure.
2 = a constant parameter (cannot vary as the design evolves),
    typically a control integer or a material property, but not a
    load, allowable, or factor of safety, which are asked for later.
3 = a parameter characterizing the environment, such as a load
    component or a temperature.
4 = a quantity that describes the response of the structure,
    (e.g. stress, buckling load, frequency)
5 = an allowable, such as maximum allowable stress, minimum
    allowable frequency, etc.
6 = a factor of safety
7 = the design objective (e.g. weight)

Please provide these data names, definitions, and ''help''
paragraphs in the following order: all variables with Roles 1 or 2
first; then all Role 3 variables; then variables with Roles 4, 5 and
6 in the repeating pattern (A4, A5, A6);  (B4, B5, B6);  (C4, C5,
C6); . . . The last variable name and definition you supply is the
single Role 7 variable: the design objective.

Next, write/gather FORTRAN code that predicts behavior such as
stress, buckling, vibration frequency, displacement, clearance, etc.
as functions of the data names that you provided via GENTEXT. The
types of behavior for which algorithms must be supplied were chosen
```

Table 1—*Continued*

```
by you during the interactive ''GENTEXT'' session. Insert this
FORTRAN code into libraries called BEHAVIOR.NEW and/or STRUCT.NEW and
ADDCODEn.NEW.  Skeletal forms of BEHAVIOR.NEW and STRUCT.NEW are
created by GENTEXT.  You may create new libraries called
ADDCODEn.NEW, n = 2, 3,..., if you think this is a good idea. This is
done in the complex example case, Example 2, documented in detail in
the file called PANEL.CAS in [10]. In order to learn more about
BEHAVIOR.NEW, STRUCT.NEW, ADDCODEn.NEW, use the ''HELPG'' command.
Information about these libraries appears on your screen when you
type one or more of the following three-word commands:

              HELPG LIBRARIES GENOPTUSER
              HELPG LIBRARIES BEHAVIOR
              HELPG LIBRARIES STRUCT
              HELPG LIBRARIES ADDCODEn.

Creating and properly installing new FORTRAN code for predicting the
GENOPT-user-defined behavior that might constrain the design is the
most challenging part of the GENOPT user's activity. When you have
completed this task, type the command GENPROGRAMS.

$ GENPROGRAMS

The GENOPT system generates object libraries called: BEGIN.OLB,
STOGET.OLB, STRUCT.OLB, BEHAVIOR.OLB, CHANGE.OLB and absolute
elements called: BEGIN.EXE, DECIDE.EXE, MAINSETUP.EXE, OPTIMIZE.EXE,
CHANGE.EXE, STORE.EXE, CHOOSEPLOT.EXE, DIPLOT.EXE,  that incorporate
the code that you wrote/gathered as described above.

End of Section 1: The user-friendly program system has been
generated.

-------------------------------------------------------------------

 SECTION 2     (TASKS PERFORMED BY THE END USER: APPLY THE NEW
                ANALYSIS/OPTIMIZATION PROGRAM TO A SPECIFIC CASE.)

$ BEGIN
You, or an end user appointed by you,  assign a specific case name
which is different from the generic case name.  Provide a starting
design, material properties, tables and control integers, loads,
allowables, and factors of safety for the specific case.

$ DECIDE
Choose decision variables and their lower and upper bounds,
linked variables and their linking coefficients, inequality
constraints based on system dimensions rather than system behavior,
and ''escape'' variables.

$ MAINSETUP
Choose the type of analysis (1=optimization, 2=analysis of a
fixed design) and the maximum number of design iterations to be
performed by OPTIMIZE.

$ OPTIMIZE
Launch a batch run that does the work specified by MAINSETUP.
Upon completion of the batch run and inspection of the results
```

Table 1—*Continued*

```
==========================================================================
provided in the file called  NAME.OPM, decide whether to do more
design iterations.  NAME = end-user-assigned specific case name:
   NAME = PLATE1 in Example 1, Part 2;
   NAME = ARIANE in Example 2, Part 2.

$ OPTIMIZE
You launch a second batch run for another set of design iterations.
The starting design for this run is the ending design of the previous
run.

$ CHOOSEPLOT
Choose which design variables and design margins should be
plotted v. design iterations.

$ DIPLOT
Cause the variables chosen by you in CHOOSEPLOT to be plotted.
==========================================================================
NOTE: The runstream given here is very similar that used in Example 1.
      The only difference is that in Example 1 only one ''OPTIMIZE"
      was needed, as convergence to an optimum design was achieved
      within the maximum number of iterations permitted in that run.
      More details appear in the file PLATE.CAS in [10].
```

type of behavior and contingency be accounted for in order that generation of impractical designs be avoided. The writer has encountered examples in which failure to characterize the behavior completely has led to disappearance of the object as optimization iterations proceed!

(3) Decide what the objective of the optimization is. In the aerospace industry it is often weight. In the civil engineering industry it is often cost. There are of course many other possible objectives. The author has encountered a problem in which the objective was minimum heat flow (Bushnell, 1988). In another problem the objective was minimum root-mean-squared error of the surface of a mirror (Bushnell, 1983a). A mixed objective is possible, with weights assigned to each component of the objective.

(4) Organize the input data. For example, are pointers needed? Should a certain class of input items be an array or should each member of this class be a simple variable with its own unique name? If a variable is an array, should it be a one-dimensional or a two-dimensional array? In what order should the user be prompted for the input data? (GENOPT determines this to some extent, as described in Section 1 of Table 1.)

(5) Choose (a) a meaningful name, (b) a clear one-line definition, and, if needed, (c) a supporting "help" paragraph or paragraphs for each input datum. Careful attention to these items will contribute greatly to the user-friendliness of the program system generated by GENOPT, since the data names and definitions appear in the output from this system seen by the end user.

(6) Write or "borrow" computer algorithms to calculate the various behaviors to be included as possible constraints on the design. In this paper two examples are presented: a simple one in which short algorithms are written, and a more complex one in which subroutines are "borrowed" from elsewhere and either used directly or modified and then used.

(7) Test the program system generated by GENOPT. Does the code introduced in Item 6 work properly? Are enough behaviors included as constraining influences on the design? Are the data names informative and the data definitions complete enough? Is there sufficient "help" documentation for a naive end user? If the end user looks at the output 6 months or a year from now will he or she be able to understand it at once?

(8) Interact with the end user. If it proves that additional types of behavior must be included, incorporate them using the "INSERT" command. ["INSERT" is described in Bushnell (1989)].

Role of the end user

Whereas the GENOPT user provides the framework within which a certain class of objects may be optimized, the end user applies the program system thus generated. In his or her applications, the end user must:

(1) Choose a specific problem that fits within the generic class established by the GENOPT user. For example, it will not do to try to optimize something in which creep is an important phenomenon if creep is not part of the "universe" conceived by the GENOPT user.

(2) Choose a starting design with appropriate sets of loads and an allowable factor of safety for each behavioral constraint. (This is done in the "BEGIN" processor, to be described below.)

(3) Choose appropriate decision variables with reasonable lower and upper bounds. (This is done in the "DECIDE" processor, to be described below.)

(4) Choose linked variables and linking expressions (equality constraints), if any. (This is done in the "DECIDE" processor.)

(5) Choose inequality constraint expressions, if any. (This is done in the "DECIDE" processor.)

(6) Choose "escape" variables. These are variables that, when increased, force the design toward the feasible region. They are needed sometimes because ADS may fail to modify a grossly unfeasible design. (This is done in "DECIDE".)

(7) During optimization use enough restarts, iterations, and "CHANGE" commands to ensure that a global optimum design has been obtained, not merely a local optimum. (The "CHANGE" command will be introduced later.)

(8) Interact with the GENOPT user. Suggest introduction of new types of behavior, better data names, definitions, and "help" paragraphs.

(9) Confirm the appropriateness of the optimum design by detailed analysis with a general purpose program and/or experiments.

PROPERTIES OF GENOPT

The GENOPT program system has the following properties:

(1) GENOPT is a FORTRAN program that writes other FORTRAN programs. These programs, when augmented by GENOPT user-supplied coding, form a menu-driven program system that is user-friendly in the GENOPT user's field. In the two examples, (a) and (b), discussed here the GENOPT user (the author in this case) had to supply FORTRAN coding that:

(a) calculates stress, buckling, vibration and displacement of rectangular, isotropic flat plates under in-plane loads or pressure, and
(b) calculates stress and various kinds of buckling for laminated, composite, ring and stringer stiffened plates and cylindrical panels and shells under multiple sets of combined in-plane loading.

(2) An analysis of a fixed design is "automatically" converted by GENOPT into the capability of optimizing that design concept should the end user wish to do so.

(3) The analysis/optimization problems need not be in fields, nor in jargon familiar to the developer of GENOPT. Although both example cases are in the field of structural analysis, GENOPT is not limited to that field.

(4) Input data and textual material are elicited from the GENOPT user in a general enough way, so that he or she can employ whatever data names, definitions and "help" paragraphs will make subsequent use of the program system thus generated easy, by those less familiar with the generic class of problems than the GENOPT user. The data names

and definitions appear in the output of the GENOPT-generated program system to be used by the end user. If meaningful data names, and clear definitions and "help" paragraphs are supplied by the GENOPT user, the analysis/optimization capability generated by GENOPT will be user-friendly for the end user.

(5) The program system generated by GENOPT has the same general architecture as previous programs written for specific applications (Bushnell, 1983a, b, 1986a, b, 1987, 1988). It is driven by a pre-established menu of commands. A typical runstream executed by the end user of the GENOPT-generated program system is:

BEGIN	(End user supplies starting design, control integers, material properties, loads, allowables, and factors of safety in an interactive- "help" mode.)
DECIDE	(End user chooses decision variables, lower and upper bounds of the decision variables, linked variables, inequality constraints based on system dimensions rather than system behavior, and "escape" variables.)
MAINSETUP	(End user chooses output option, whether to perform an analysis of a fixed design or to optimize, and the number of design iterations.)
OPTIMIZE	(The program system performs, in a batch mode, the work specified in MAINSETUP.)
CHOOSEPLOT	(End user chooses which variables to plot versus design iterations.)
DIPLOT	(End user obtains plots.)

ARCHITECTURE OF THE GENOPT MAINPROCESSOR AND OPTIMIZATION STRATEGY

Each command "OPTIMIZE", which occurs twice in Section 2 of Table 1, causes a batch run of the GENOPT mainprocessor to be launched. The architecture of this mainprocessor, which is called "MAIN", is depicted in Fig. 1. It is assumed in Fig. 1 that the decision variables have been stored in a vector **X** of length NDV. The behavior of the system being optimized is determined once in SUBROUTINE STRUCT for the unperturbed design and NDV times, again in SUBROUTINE STRUCT, for the design in which each of the decision variables $X(i)$ is perturbed in turn by 5% [$dX(i) = 0.05*\text{ABS}(X(i))$]. Gradients of the behavioral constraints are calculated, yielding a matrix of numbers with dimensions NCONST × NDV, in which NCONST is the number of behavioral constraints and NDV is the number of decision variables. This information is provided in a scaled form as input

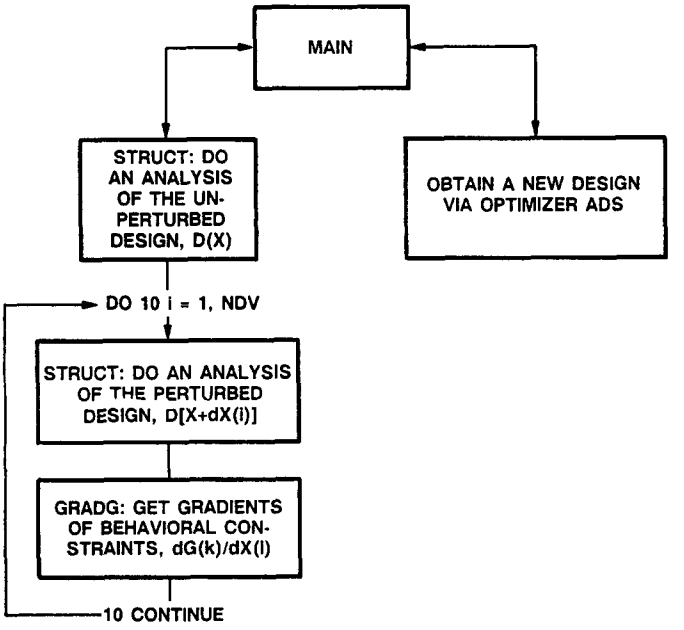

Fig. 1. Architecture of the GENOPT mainprocessor "MAIN".

to ADS, which derives a new design based on the modified method of feasible directions (Vanderplaats, 1987; Vanderplaats and Sugimato, 1986). The new design derived by ADS is used as a new unperturbed design on the left-hand side of Fig. 1. Calculations proceed in this manner until ADS can do no more or until the maximum number of iterations specified by the end user in MAINSETUP has been reached. Further iterations are performed by successive commands "OPTIMIZE", as listed in Tables 1 and 10.

Section 2 of Table 1 shows the command "OPTIMIZE" being typed twice. Each "OPTIMIZE" run was "instructed" by MAINSETUP to permit a maximum of n design iterations. One might well ask, "Why take two sets of n iterations each, instead of just one set with $2n$ or more iterations?" There are two reasons. The first is that it is always a good idea for the user to monitor carefully the output as the design evolves. Hence, after each set of n iterations the user should inspect the results.

The second reason has to do with how the optimizer ADS works in GENOPT. Figure 2 illustrates the case of two decision variables. Given the user's lower and upper bounds of the decision variables, GENOPT establishes a more restrictive "window" within which the decision variables are allowed to vary in any given iteration. The size of this "window" decreases with each design iteration by a factor of 0.8. These restrictions are placed upon the permitted excursions of the decision variables in order to prevent wild swings in the design as it evolves. The example shown in Fig. 2 shows eight iterations, and the last design point is labeled "FINAL DESIGN FOR THIS SET OF ITERATIONS". Note that the final "window" of permitted excursions is rather small. To the user it may appear that design iterations have converged, whereas actually they may not have. On the other hand, they really may have converged. This can easily be checked simply by executing "OPTIMIZE" again. When this is done, the "window" of permitted excursions is re-expanded to its original size, the new starting design is the last design obtained in the previous set of iterations, and new iterations proceed as before. The user should keep executing "OPTI-

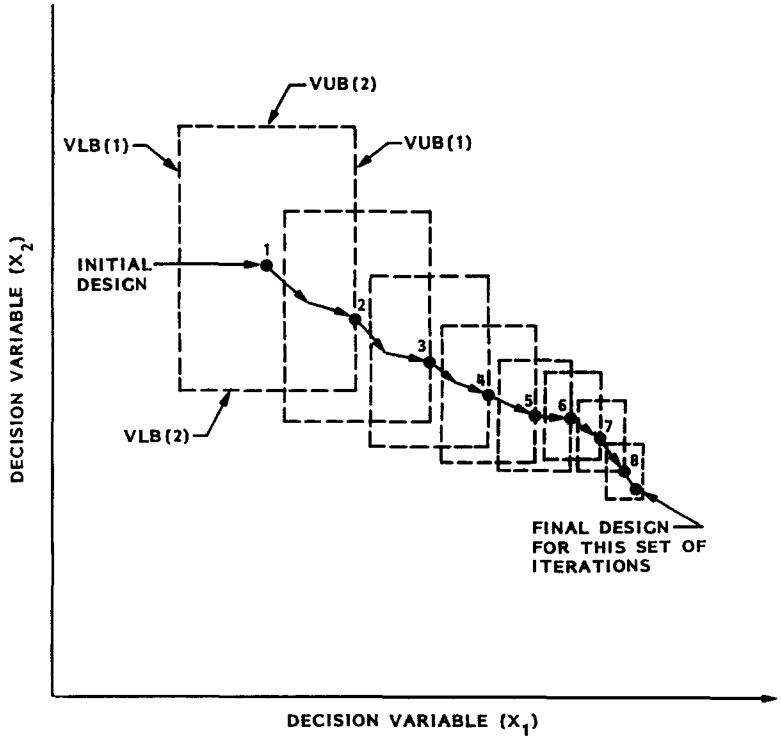

Fig. 2. Schematic of the evolution of a design with one set of design iterations and two decision variables, $X(1)$ and $X(2)$. With each iteration the optimizer, ADS, establishes a "window" of permitted excursion of the decision variables. In GENOPT this "window" shrinks by a factor of 0.8 with each design iteration. Upon re-execution of OPTIMIZE the "window" is re-expanded to its original size, which depends upon lower and upper bounds supplied by the end user in DECIDE and upon certain strategies used by ADS.

MIZE" until the objective does not change very much. When the user is satisfied with the current design, he or she should then take one additional set of design iterations in which the number of iterations is larger than the number used previously, say twice or three times as many.

EXAMPLE 1: MINIMUM WEIGHT DESIGN OF A PLATE

Part 1: Tasks performed by the GENOPT user

Statement of the problem. This case is a bit contrived. However, it demonstrates a relatively simple use of GENOPT. The problem is to find the minimum weight design of an isotropic, rectangular flat plate simply supported along all four edges. The plate is subject to the following environments, each applied as if it were a separate "event" that occurs during operation of the object of which the plate is a part:

(1) uniform axial compression,
(2) uniform in-plane shear,
(3) uniform normal pressure.

These separate loadings might represent environments seen by the structure during different phases of its operation.

The following behavioral constraints apply:

(1) the maximum effective stress in the plate shall not exceed an allowable value to be specified by the end user;
(2) the plate is not allowed to buckle under the axial compression or under the in-plane shear loads to be specified by the end user;
(3) the minimum natural frequency of the unloaded plate must be greater than an allowable value to be specified by the end user;
(4) the maximum normal deflection of the plate under uniform normal pressure to be specified by the end user shall not exceed an allowable value to be specified by the end user.

The following constraints on the geometry of the plate will be imposed by the end user in the interactive "DECIDE" session:

(1) the area of the plate shall not exceed 100 (units)**2;
(2) the area of the plate shall not be less than 50 (units)**2;
(3) the variable LENGTH (to be defined later as "length of the plate") shall be greater than the variable WIDTH (to be defined later as "width of the plate").

This problem is defined and solved by a runstream similar to that given in Table 1.

Preparation for the "GENTEXT" interactive session: (1) Finding models for predicting the behavior of the plate:

An important task in solving the problem just stated is to find predictors for the four behavioral constraints listed above. These predictors can be found in handbooks. In this case Roark (1954) has formulas for buckling of rectangular, isotropic, simply supported plates under uniform axial compression (Table XVI, Entry No. 1) and under uniform in-plane shear (Table XVI, Entry No. 10). For uniform axial compression the critical axial stress resultant Nx(cr) (stress resultant Nx = stress*thickness) is given by:

$$Nx(\text{cr}) = K*[E*t/(1-nu**2)]*(t/b)**2 \tag{1}$$

in which K is a coefficient that depends on the length/width ratio a/b of the plate; E is the elastic modulus of the plate material; t is the plate thickness; nu is the material Poisson ratio; and b is the width of the plate. Roark gives a table of K values versus a/b. The critical in-plane shear resultant Nxy(cr) has the same form as eqn (1). The coefficient K depends on a/b in a different way from that for axial compression. Roark gives a table of K versus a/b for shear buckling in Entry No. 10 of Table XVI. Since the Roark formulas are to be

used in this case, the GENOPT user will have to introduce arrays of constant parameters:

(1) $K1(i)$ and ALPHA1(i) for axial compression buckling coefficients;
(2) $K2(j)$ and ALPHA2(j) for in-plane shear buckling coefficients,

in which $K1$ and $K2$ represent K in eqn (1), and ALPHA1 and ALPHA2 represents the plate aspect ratios a/b for which Roark gives K in Entries 1 and 10, respectively, of Table XVI in Roark (1954). The GENOPT user will also have to use an interpolating routine to calculate K for values of a/b between the discrete values given in Roark. In this case a subroutine called INTERP is used. [See Table 6(b).]

The algorithms for compression and shear buckling appear in Table 6(b). The GENOPT user must set the maximum permitted values of the indices i and j large enough to give an end user reasonable freedom to use tabular data from any source, perhaps from design curves.

The discussion on the axial and shear buckling behaviors is an important ingredient of this example because it illustrates how design curves would be used as input to GENOPT for automated optimization of systems for which graphic or tabular design data are available.

Table 2. Seven roles of GENOPT input data and data names chosen by the GENOPT user for Example 1

ROLE OF DATA	DATA NAMES	SHORT DEFINITIONS
1. Candidates for decision variables	THICK, LENGTH, WIDTH	thickness t, length a, width b.
2. Constant parameters	E, NU, RHO	modulus, Poisson ratio, weight density
	AOBAXL(i), KAXL(i)	a/b, K values from Roark for compression buckling
	AOBSHR(j), KSHR(j)	a/b, K values from Roark for shear buckling.
3. Environmental parameters (loads)	Nx(k), Ny(k), Nxy(k), PRESS(k)	axial resultant, transverse resultant, in-plane shear resultant, normal pressure for the kth load set.
4. Responses (behavioral constraints)	STRESS(k), BUCKLE(k), FREQ(k), W(k)	max. stress, buckling load factor, natural frequency, max. normal displacement for the kth load set.
5. Allowables	MAXSTR(k), MINBUC(k), MINCPS(k), AW(k)	allowable stress, buckling load factor, frequency, normal displacement for the kth load set.
6. Factors of safety	FSTRES(k), FBUCKL(k), FSFREQ(k), FW(k)	factors of safety for stress, buckling, frequency, displacement for the kth load set.
7. Objective function	WEIGHT	weight of the plate.

Roark (1954) also provides formulas for the maximum stress and normal deflection of rectangular, isotropic, simply supported plates under uniform normal pressure (Table X, Entry No. 36). Formulas for the maximum stress appear in Table 6(a) and for maximum normal deflection in Table 6(d).

Leissa (1969) provides a formula for the fundamental vibration frequency of a simply supported, isotropic, rectangular plate. The formula appears in Table 6(c).

The objective function, WEIGHT, is simply

$$\text{WEIGHT} = (\text{weight density}) * a * b * t \tag{2}$$

(2) Organizing and deciding on names for the input data:

GENTEXT forces the GENOPT user to classify generic problem data into seven categories or roles. Table 2 lists these roles and shows the data names that the GENOPT user decided to use in this example.

The "GENTEXT" interactive session: The following is a list (abridged to save space) of part of the interactive session initiated when the GENOPT user types the command "GENTEXT". Input from the user is set in boldface.

```
********** BEGINNING OF INTERACTIVE ''GENTEXT'' SESSION **********
$ GENTEXT
    ENTER THE GENERIC CASE NAME: PLATE

ARE YOU CORRECTING, ADDING TO, OR USING PLATE.INP ? (y or n):  n

You have chosen the following name for this case:  PLATE.
Henceforth, this will be called the ''generic'' name. After you
have completed your tasks in GENOPT, you and other users
will be able to optimize specific things that fit within the
class of optimization problems that you have called PLATE.
CHOOSE STARTING PROMPT INDEX (integer from 1 to 10.  Try 5):  5
CHOOSE INCREMENT FOR PROMPT INDEX (integer from 1 to 10.  Try 5):  5

CHOOSE: 0  or  1:
  (0 means ''introductory explanatory text'')
  (1 means ''one-line input datum prompt'')
CHOOSE: 0 or 1:  1

You will next be asked to provide information
about a new variable that will play a role in your program.
The following items relative to this new variable will be
asked of you:
  1. A name of the variable (six characters or less).
  2. The role of the variable in your program.
  3. Is the variable an array? (If yes, give number of rows,
     NROWS and columns, NCOLS.)
  4. A one-line definition of the variable.
  5. Do you want to include a ''help'' paragraph that explains
     more about the variable than the one-line definition?
  6. If you answer 5. with Y, you provide a help paragraph.

The variable can have one of the following roles:
  1 = a possible decision variable for optimization,
      typically a dimension of a structure.
  2 = a constant parameter (cannot vary as the design evolves),
      typically a control integer or a material property,
```

```
      but not a load, allowable, or factor of safety,
      which are asked for later.
  3 = a parameter characterizing the environment, such
      as a load component or a temperature.
  4 = a quantity that describes the response of the
      structure, (e.g. stress, buckling load, frequency)
  5 = an allowable, such as maximum allowable stress,
      minimum allowable frequency, etc.
  6 = a factor of safety
  7 = the quantity that is to be minimized (e.g. weight)

        (LINES SKIPPED IN ORDER TO SAVE SPACE.)

PROVIDE A NAME FOR THE VARIABLE (6 or less characters, CAPS): THICK
IDENTIFY ROLE OF THICK (1 or 2 or 3 or 4 or 5 or 6 or 7):  1
Is the variable THICK an array?  n
PROVIDE A DEFINITION FOR THICK. (LESS THAN 60 CHARACTERS!)
thickness of the plate
THICK = thickness of the plate
DO YOU WANT TO INCLUDE A ''HELP'' PARAGRAPH? (y or n):  n
ANY MORE DECISION VARIABLE CANDIDATES (ROLE 1 VARIABLES)
OR FIXED PARAMETERS (e.g.  material) (ROLE 2 VARIABLES)? y

      (MANY LINES SKIPPED IN ORDER TO SAVE SPACE)

CHOOSE AN ENVIRONMENTAL PARAMETER (load)   (ROLE 3 VARIABLE).
CHOOSE: 0=introductory text or 1=one-line input datum prompt:  0
PROVIDE INTRODUCTORY EXPLANATORY TEXT
NEXT, PROVIDE ALL ENVIRONMENTAL PARAMETERS (LOADS)
ANY MORE LINES IN THIS PARAGRAPH? (y or <cr> or n):  n
CHOOSE: 0=introductory text or 1=one-line input datum prompt:  1
PROVIDE A NAME FOR THE VARIABLE (6 or less characters, CAPS): Nx
IDENTIFY ROLE OF Nx (1 or 2 or 3 or 4 or 5 or 6 or 7):  3
PROVIDE A DEFINITION FOR Nx.  (LESS THAN 60 CHARACTERS!)
Axial tension per unit width of the plate (lb/in)
Nx = Axial tension per unit width of the plate (lb/in)
DO YOU WANT TO INCLUDE A ''HELP'' PARAGRAPH? (y or n):  y
PROVIDE HELP PARAGRAPH TO EXPLAIN INPUT
NOTE: Nx must be negative for axial compression!
ANY MORE LINES IN THIS PARAGRAPH? (y or <cr> or n):  n
ANY MORE ENVIRONMENTAL PARAMETERS (loads) (ROLE 3 VARIABLES)? y

      (MANY LINES SKIPPED IN ORDER TO SAVE SPACE)

CHOOSE A RESPONSE PARAMETER (e.g. stress)  (ROLE 4 VARIABLE).
CHOOSE: 0=introductory text or 1=one-line input datum prompt:  1
```

```
PROVIDE A NAME FOR THE VARIABLE (6 or less characters, CAPS): FREQ
IDENTIFY ROLE OF FREQ (1 or 2 or 3 or 4 or 5 or 6 or 7):  4

FREQ is an array with the number of rows equal to 20 .
  Each row corresponds to a load case. Max. of 20 load cases permitted.
  Do you want to reset the number of columns in FREQ?
  In this case, if you answer ''no'' FREQ would be dimensioned FREQ(20).
Do you want to reset the number of columns in FREQ? n
PROVIDE A DEFINITION FOR FREQ. (LESS THAN 60 CHARACTERS!)
Fundamental frequency of unloaded plate
FREQ = Fundamental frequency of unloaded plate
DO YOU WANT TO INCLUDE A ''HELP'' PARAGRAPH? (y or n):  n

CHOOSE AN ALLOWABLE FOR FREQ (ROLE 5 VARIABLE).
CHOOSE: 0=introductory text or 1=one-line input datum prompt:  1
PROVIDE A NAME FOR THE VARIABLE (6 or less characters, CAPS): MINCPS
PROVIDE A DEFINITION FOR MINCPS. (LESS THAN 60 CHARACTERS !)
Minimum allowable value for the fundamental frequency
MINCPS = Minimum allowable value for the fundamental frequency
DO YOU WANT TO INCLUDE A ''HELP'' PARAGRAPH? (y or n):  n

CHOOSE FACTOR OF SAFETY FOR FREQ (ROLE 6 VARIABLE).
CHOOSE: 0=introductory text or 1=one-line input datum prompt:  1
PROVIDE A NAME FOR THE VARIABLE (6 or less characters, CAPS): FSFREQ
PROVIDE A DEFINITION FOR FSFREQ. (LESS THAN 60 CHARACTERS !)
Factor of safety for FREQ
FSFREQ = Factor of safety for FREQ
DO YOU WANT TO INCLUDE A ''HELP'' PARAGRAPH? (y or n):  n


 (MANY LINES SKIPPED IN ORDER TO SAVE SPACE)


ANY MORE RESPONSE VARIABLES (e.g.buckling) (ROLE 4 VARIABLES)? n

CHOOSE AN OBJECTIVE (e.g.  minimum weight) (ROLE 7 VARIABLE).
CHOOSE: 0=introductory text or 1=one-line input datum prompt:  0
PROVIDE INTRODUCTORY EXPLANATORY TEXT
LAST, AN OBJECTIVE MUST BE CHOSEN, SUCH AS MINIMUM WEIGHT
ANY MORE LINES IN THIS PARAGRAPH? (y or <cr> or n):  y
OR MINIMUM COST.
ANY MORE LINES IN THIS PARAGRAPH? (y or <cr> or n):  n
CHOOSE: 0=introductory text or 1=one-line input datum prompt:  1
PROVIDE A NAME FOR THE VARIABLE (6 or less characters, CAPS): WEIGHT
PROVIDE A DEFINITION FOR WEIGHT. (LESS THAN 60 CHARACTERS !)
Weight of the plate
WEIGHT = Weight of the plate
DO YOU WANT TO INCLUDE A ''HELP'' PARAGRAPH? (y or n):  n


  DUMMY ENTRY WRITTEN AT END OF PLATE.PRO

  STOGET.NEW HAS BEEN CREATED.
  BEGIN.NEW HAS BEEN CREATED.
  STRUCT.NEW HAS BEEN CREATED.
  BEHAVIOR.NEW HAS BEEN CREATED.
  CHANGE.NEW HAS BEEN CREATED.

********** END OF THE ''GENTEXT'' INTERACTIVE SESSION **********
```

Files produced by the interactive "GENTEXT" session. The interactive "GENTEXT" session produces many files, all of which are described in Bushnell (1989) and in the interactive "help" utility called "HELPG" that forms part of the GENOPT system. In order to save space only a few of these files will be described here.

One of the most important files is called NAME.PRO, in which NAME is the generic name selected by the GENOPT user (NAME = PLATE in this example). The file PLATE.PRO is listed in Table 3. PLATE.PRO is used by the "BEGIN" processor, which will be executed by the end user. Note that while the prompt numbers and format are established by GENOPT, the text (except for certain array dimension names) is created by the GENOPT user. This text, particularly the one-line definitions given on the lines in Table

Table 3. List of the file PLATE.PRO which provides the end user with prompts and "HELPS" when the end user uses "BEGIN"

```
5.0
     PROGRAM FOR OPTIMIZATION OF A RECTANGULAR PLATE
     SUBJECTED TO SEVERAL LOADING ENVIRONMENTS AND
     CONSTRAINTS ON STRESS, BUCKLING, DISPLACEMENT, AND FREQUENCY.


10.0
     FIRST, PROVIDE ALL VARIABLES THAT CAN BE DECISION VARIABLES,
     THAT IS, VARIABLES THAT CAN CHANGE DURING OPTIMIZATION
     ITERATIONS (ROLE TYPE 1), AND FIXED VARIABLES (ROLE TYPE 2).

15.1 thickness of the plate: THICK
20.1 Length of the plate: LENGTH
25.1 Width of the plate: WIDTH
30.1 Young's modulus of the plate material: E
35.1 Poisson's ratio of the plate material: NU
40.1 Weight density (e.g. lb/in**3) of the plate material: RHO

45.0
     THE FOLLOWING TABLE (FROM ROARK, 3RD EDITION, TABLE XVI,
     ENTRY NO. 1, PAGE 312), GIVES THE RELATIONSHIP OF PLATE
     (LENGTH/WIDTH) TO A COEFFICIENT FOR BUCKLING UNDER UNIFORM
     AXIAL COMPRESSION.

50.1 Number IAOBAX of rows in the array  AOBAXL: IAOBAX
55.1 (plate length, LENGTH)/(plate width, WIDTH): AOBAXL
60.1 Coefficient for buckling under uniform axial compression: KAXL

65.0
     THE FOLLOWING TABLE (FROM ROARK, 3RD EDITION, TABLE XVI,
     ENTRY NO. 10, PAGE 312), GIVES THE RELATIONSHIP OF PLATE
     (LENGTH/WIDTH) TO A COEFFICIENT FOR BUCKLING UNDER UNIFORM
     IN-PLANE SHEAR.

70.1 Number IAOBSH   of rows in the array  AOBSHR: IAOBSH
75.1 (plate length, LENGTH)/(plate width, WIDTH): AOBSHR
80.1 Coefficient for buckling under uniform in-plane shear: KSHR

85.0
     NEXT, PROVIDE ALL ENVIRONMENTAL PARAMETERS (LOADS, TEMPERATURES)

90.1 Number NCASES of load cases (environments): NCASES
95.1 Axial tension per unit width of the plate (lb/in): Nx
```

Table 3—*Continued*

```
 95.2
      NOTE:  Nx must be negative for axial compression!

100.1 Transverse tension per unit length of the plate (lb/in): Ny
105.1 In-plane shear per unit edge length applied to the plate.: Nxy
110.1 Uniform normal pressure on the plate: PRESS

115.0
      NEXT, PROVIDE RESPONSE PARAMETERS, ALLOWABLES, AND FACTORS OF
      SAFETY. THE ORDER IN WHICH YOU MUST PROVIDE THESE DATA IS:
      A,B,C; A,B,C; A,B,C; A,B,C; etc, in which
      A = RESPONSE,   B = ALLOWABLE,   C = FACTOR OF SAFETY.

120.0 Maximum effective (von Mises) stress: STRESS
125.1 Maximum effective (von Mises) stress allowed: MAXSTR
130.1 Factor of safety for effective stress: FSTRES
135.0 Buckling load factor: BUCKLE
140.1 Minimum allowable buckling load factor (use 1.0): MINBUC
145.1 Factor of safety for buckling load factor: FBUCKL
150.0 Fundamental frequency of unloaded plate: FREQ
155.1 Minimum allowable value for the fundamental frequency: MINCPS
160.1 Factor of safety for FREQ: FSFREQ
165.0 Normal deflection under uniform pressure: W
170.1 Maximum allowable normal deflection under pressure: AW
175.1 Factor of safety for max deflection under pressure: FW

180.0
      LAST, AN OBJECTIVE MUST BE CHOSEN, SUCH AS MINIMUM WEIGHT
      OR MINIMUM COST.

185.0 Weight of the plate: WEIGHT
999.0 DUMMY ENTRY TO MARK END OF FILE
```

```
NOTE: The GENOPT user provides the starting prompt index and index
increment, the names of the variables, and the text. GENTEXT provides
the prompt numbers and the layout of the prompt file. Informative
data names and clear text will make the GENTEXT- generated
optimization program system easy to use.
```

3 identified with indices of the form xx.1, will appear throughout the output from the system of programs (BEGIN, DECIDE, OPTIMIZE, CHANGE, CHOOSEPLOT) to be used by the end user. If the GENOPT user composes clear one-line prompts and frequent, well-written "help" paragraphs, the GENTEXT-generated optimization program system will be end user-friendly.

During the interactive "GENTEXT" session various blocks of FORTRAN code are written by the GENOPT system. Examples are listed in Tables 4 and 5. This machine-generated FORTRAN code is concatenated with certain other files that contain FORTRAN fragments to form FORTRAN libraries called BEGIN.NEW, STOGET.NEW, STRUCT.NEW, BEHAVIOR.NEW, and CHANGE.NEW, as noted by GENOPT in the last five lines of the interactive "GENTEXT" session listed above. These libraries are described in Bushnell (1989). Only BEHAVIOR.NEW and STRUCT.NEW may be modified by the GENOPT user.

In this example, only BEHAVIOR.NEW is modified by the GENOPT user. Table 5 lists BEHAVIOR.NEW in its skeletal form. Table 6 lists the algorithms created by the

Table 4. Part of the file PLATE.CON generated by "GENTEXT", this file contains calls to BEHX1, BEHX2, ... BEHX*n* and to subroutine CONX, which generates behavioral constraints for optimization. This file forms part of STRUCT.NEW

```
=================================================================
C
C  Behavior and constraints generated next for STRESS:
C  STRESS = Maximum effective (von Mises) stress
C
      PHRASE =
     1 'Maximum effective (von Mises) stress'
      CALL BLANKX(PHRASE,IENDP4)
      CALL BEHX1  (IFILE8,NPRINX,IMODX, ILOADX
     1 'Maximum effective (von Mises) stress')
      IF (STRESS(ILOADX  ).EQ.0.)  STRESS(ILOADX  ) = 1.E-10
      IF (MAXSTR(ILOADX  ).EQ.0.)  MAXSTR(ILOADX  ) = 1.0
      IF (FSTRES(ILOADX  ).EQ.0.)  FSTRES(ILOADX  ) = 1.0
      KCONX = KCONX + 1
      CARX(KCONX) =STRESS(ILOADX  )
      WORDCX= 'MAXSTR('//CIX//')/[STRESS('//CIX//
     1  ') X FSTRES('//CIX//')]'
      CALL CONX(STRESS(ILOADX  ),MAXSTR(ILOADX  ),FSTRES(ILOADX  )
     1,'Maximum effective (von Mises) stress',
     1 'Maximum effective (von Mises) stress allowed',
     1 'Factor of safety for effective stress',
     1 1,INUMTT,IMODX,CONMAX,ICONSX,IPOINC,CONSTX,WORDCX,
     1 WORDMX,PCWORD,CPLOTX,ICARX)
      IF (IMODX.EQ.0) THEN
         CODPHR =
     1    Maximum effective (von Mises) stress:
         IENDP4 =40
         CODNAM ='STRESS('//CIX//')'
         MLET4 =6 + 4
         WORDBX(KCONX)= CODPHR(1:IENDP4)//CODNAM(1:MLET4)
         IF (NPRINX.GT.0) WRITE(IFILE8,'(I5,6X,G14.7,A,A)')
     1    KCONX,CARX(KCONX),CODPHR(1:IENDP4),CODNAM(1:MLET4)
      ENDIF
  130 CONTINUE
  131 CONTINUE

C-----------------------------------------------------------------
C
C  Behavior and constraints generated next for BUCKLE:
C  BUCKLE = Buckling load factor
C
    (LINES OF CODE OMITTED IN THE FOLLOWING IN ORDER TO SAVE SPACE)

      PHRASE = 'Buckling load factor'
      CALL BEHX2  (IFILE8,NPRINX,IMODX, ILOADX, 'Buckling load factor')
      WORDCX= 'BUCKLE('//CIX//')/[MINBUC('//CIX//') X FBUCKL('//CIX//')]
      CALL CONX(BUCKLE(ILOADX  ),MINBUC(ILOADX  ),FBUCKL(ILOADX  ) . . .

C-----------------------------------------------------------------
C
C  Behavior and constraints generated next for FREQ:
C  FREQ = Fundamental frequency of unloaded plate
C
    (LINES OF CODE OMITTED IN THE FOLLOWING IN ORDER TO SAVE SPACE)

      PHRASE = 'Fundamental frequency of unloaded plate'
      CALL BEHX3(IFILE8,NPRINX,IMODX,ILOADX,'Fundamental frequency. . .
```

Table 4—*Continued*

```
      WORDCX= 'FREQ('//CIX//')/[MINCPS('//CIX//') X FSFREQ('//CIX//')]'
      CALL CONX(FREQ(ILOADX  ),MINCPS(ILOADX  ),FSFREQ(ILOADX  ). . . .
C-----------------------------------------------------------------
C
C  Behavior and constraints generated next for W:
C  W = Normal deflection under uniform pressure
C
    (LINES OF CODE OMITTED IN THE FOLLOWING IN ORDER TO SAVE SPACE)

      PHRASE = 'Normal deflection under uniform pressure'
      CALL BEHX4(IFILE8,NPRINX,IMODX,ILOADX,'Normal deflection under. . .
      WORDCX= 'AW('//CIX//')/[W('//CIX//') X FW('//CIX//')]'
      CALL CONX(W(ILOADX  ),AW(ILOADX  ),FW(ILOADX  ) . . . . . .
C-----------------------------------------------------------------
 1000 CONTINUE
C  END OF LOOP OVER NUMBER OF LOAD SETS (environments)
C
C  NEXT, EVALUATE THE OBJECTIVE, OBJGEN:
      PHRASE ='Weight of the plate'
      CALL BLANKX(PHRASE,IENDP4)
      CALL OBJECT(OBJGEN,'Weight of the plate')
      NCONSX = ICONSX
      RETURN
      END
C
C  End of the final portion of STRUCT written by "GENTEXT"
=====================================================================
NOTES:
(1) Skeletal SUBROUTINES BEHXn are listed in TABLE 5. Algorithms
    written by GENOPT user for this case are listed in TABLE 6.
(2) SUBROUTINE CONX is included in the GENOPT:MAIN.NEW library.
(3) SUBROUTINE OBJECT is part of the BEHAVIOR.NEW library. The
    skeletal version is listed in TABLE 5 and the particular
    algorithm for the objective for this case is listed in TABLE 6e.
(4) About ''behavior'', ''behavioral constraint'', ''margin'':
    (i) ''behavior'': refers to a quantity. Example: FREQ(ILOAD).
   (ii) ''behavioral constraint'': refers to a quotient that plays
          a role in constraining the design. The form of a ''type 1''
          constraint (see ''GENTEXT'' interactive session in [10]) is:
          (allowable)/[(behavior)*(factor of safety)].
          The form of a ''type 2'' constraint is:
          (behavior)/[(allowable)*(factor of safety)].
          EXAMPLE: MAXSTR(ILOAD)/[STRESS(ILOAD)*FSTRES(ILOAD)]  (type 1)
          EXAMPLE: FREQ(ILOAD)/[MINCPS(ILOAD)*FSFREQ(ILOAD)]    (type 2)
  (iii) ''margin'' = ''behavioral constraint'' - 1.0
```

Table 5. Part of the file BEHAVIOR.NEW generated by "GENTEXT". Skeletal routines BEHX1, BEHX2, ... BEHX*n* and subroutine OBJECT are to be completed by the GENOPT user in this case.

```
=====================================================================
C=DECK       BEHAVIOR.NEW

   (LINES OF DESCRIPTION OMITTED TO SAVE SPACE.)
   (COMMENT LINES HAVE BEEN OMITTED FROM THE BEHX* ROUTINES TO SAVE SPACE.)


C=DECK       BEHX1
      SUBROUTINE BEHX1  (IFILE,NPRINX,IMODX,ILOADX,PHRASE)
C
C   PURPOSE: OBTAIN Maximum effective (von Mises) stress
C
   (Common blocks in the file PLATE.COM are inserted here by GENOPT.)
C
       (See Table 6a for the algorithm used in ''PLATE'',
        which must be inserted here by the GENOPT user.)
C
      RETURN
      END


     (SUBROUTINES BEHX2 AND BEHX3 ARE OMITTED TO SAVE SPACE.)


C=DECK       BEHX4
      SUBROUTINE BEHX4  (IFILE,NPRINX,IMODX,ILOADX,PHRASE)
C
C   PURPOSE: OBTAIN Normal deflection under uniform pressure
C
   (Common blocks in the file PLATE.COM are inserted here by GENOPT.)
C
       (See Table 6d for the algorithm used in ''PLATE'',
        which must be inserted here by the GENOPT user.)
C
      RETURN
      END
C
C
C=DECK       OBJECT
      SUBROUTINE OBJECT(OBJGEN,PHRASE)
C   PURPOSE:Weight of the plate
C
   (Common blocks in the file PLATE.COM are inserted here by GENOPT.)
C
       (See Table 6e for the algorithm used in ''PLATE'',
        which must be inserted here by the GENOPT user.)
C
      OBJGEN =WEIGHT
C
      RETURN
      END
C
=====================================================================
NOTES:

(1) The GENOPT user completes the BEHAVIOR.NEW library by inserting
certain algorithms for predicting the behavior. The algorithms for
this example (''PLATE'')  are listed in TABLE 6.
```

Table 5—*Continued*

Notes—*Continued*

(2) In the more complex case, ''PANEL'', only SUBROUTINE OBJECT was completed. (See TABLE 28 of [10]). The behavioral characteristics were computed in new subroutines introduced by the GENOPT user into the library ADDCODE1.NEW and called from STRUCT.NEW as listed in TABLE 26 of [10]. In the more complex case ''PANEL'' the existence of previously written code made this strategy easier than using BEHX1, BEHX2,... BEHXn.

(3) Notice that the argument IMODX appears in the argument lists of the BEHX1, BEHX2,...BEHXn subroutines. It may be in some cases that calculation of the behavior for the perturbed design takes much less computer time than does calculation of this behavior for the unperturbed design. (This is true for the complex example, ''PANEL''.) The GENOPT user may want to include IMODX in his/her strategy for efficient calculation of the various behaviors for perturbed designs.

Table 6. Algorithms used in BEHAVIOR.NEW for example 1, the case called "PLATE".

(a) GENOPT-user-written code for insertion into SUBROUTINE BEHX1:

```
C   PURPOSE: OBTAIN Maximum effective (von Mises) stress
C
      SA1 = Nx(ILOADX)/THICK
      SB1 = Ny(ILOADX)/THICK
      SAB = Nxy(ILOADX)/THICK
      ALPHA = WIDTH/LENGTH
C
C  MAX. EFFECTIVE STRESS UNDER UNIFORM PRESSURE (LINEAR THEORY)
C  (TAKEN FROM ROARK, 3RD EDITION, 1954, TABLE X, FORMULA 36, P. 203
C   VALID FOR POISSON RATIO NU = 0.3) NOTE: VALID ONLY FOR LINEAR
C   BEHAVIOR!!!
C
      SA2 = (PRESS(ILOADX)*WIDTH**2/THICK**2)*(0.225 +0.382*ALPHA**2
     1                                         -0.320*ALPHA**3)
      SB2 = 0.75*PRESS(ILOADX)*WIDTH**2/
     1              (THICK**2*(1. +1.61*ALPHA**3))
C
C  EFFECTIVE STRESS AT SURFACE WHERE PRESSURE IS APPLIED:
C
      SATOP = SA1 - SA2
      SBTOP = SB1 - SB2
      SEFTOP= SQRT(SATOP**2 +SBTOP**2 - SATOP*SBTOP +3.*SAB**2)
C
C  EFFECTIVE STRESS AT OPPOSITE SURFACE:
C
      SABOT = SA1 + SA2
      SBBOT = SB1 + SB2
      SEFBOT= SQRT(SABOT**2 +SBBOT**2 - SABOT*SBBOT +3.*SAB**2)
C
      STRESS(ILOADX) = MAX(SEFTOP,SEFBOT)
```

```
(b) GENOPT-user-written code for insertion into SUBROUTINE BEHX2:
C
C   PURPOSE: OBTAIN Buckling load factor
C
      BUCKLE(ILOADX) = 0.
C
      GO TO (10,20,30), ILOADX
C
   10 CONTINUE
C
C  BUCKLING LOAD FACTOR UNDER UNIFORM AXIAL COMPRESSION, Nx:
C  TAKEN FROM ROARK, 3RD EDITION, 1954, TABLE XVI, FORMULA 1, P. 312
C
      CALL INTERP(IFILE,IAOBAX,AOBAXL,KAXL,LENGTH/WIDTH,COEFAX)
C     WRITE(6,*)' COEFAX,LENGTH/WIDTH =',COEFAX,LENGTH/WIDTH
      IF (Nx(ILOADX).LT.0.)
     1 BUCKLE(ILOADX) =(COEFAX*(E*THICK/(1.-NU**2))*(THICK/WIDTH)**2)/
     1 ABS(Nx(ILOADX))
      GO TO 50
C
   20 CONTINUE
C
C  BUCKLING UNDER UNIFORM IN-PLANE SHEAR, Nxy:
C  AGAIN, TAKEN FROM ROARK, 3RD EDITION, TABLE XVI, FORMULA 10, P. 313
C
      CALL INTERP(IFILE,IAOBSH,AOBSHR,KSHR,LENGTH/WIDTH,COEFSH)
C     WRITE(6,*)' COEFSH,LENGTH/WIDTH =',COEFSH,LENGTH/WIDTH
      IF (Nxy(ILOADX).NE.0.)
     1 BUCKLE(ILOADX)=(COEFSH*(E*THICK/(1.-NU**2))*(THICK/WIDTH)**2)/
     1 ABS(Nxy(ILOADX))
      GO TO 50
C
   30 CONTINUE
C
C  THERE IS NO BUCKLING UNDER UNIFORM NORMAL PRESSURE.
C
   50 CONTINUE
-----------------------------------------------------------------------

(c) GENOPT-user-written code for insertion into SUBROUTINE BEHX3:
C
C   PURPOSE: OBTAIN Fundamental frequency of unloaded plate
C
      FREQ(ILOADX) = 0.
C
      GO TO (50,50,30),ILOADX
C
C  FIND FUNDAMENTAL NATURAL FREQUENCY FOR UNLOADED PLATE.
C  THE VALUE ESTABLISHED FOR ILOADX = 3 IGNORES THE EFFECT OF ANY
C  TENSION THAT MAY DEVELOP IN THE PLATE WHEN IT IS LOADED BY NORMAL
C  PRESSURE.  NO FREQUENCIES ARE CALCULATED FOR THE PLATE AS
C  LOADED AXIALLY (LOAD CASE 1) OR IN UNIFORM IN-PLANE SHEAR
C  (LOAD CASE 2).
C
   30 CONTINUE
      D = E*THICK**3/(12.*(1.-NU**2))
      DMASS = RHO/386.4
      PI = 3.1415927
```

```
C
C  CALCULATE FUNDAMENTAL FREQUENCY IN CPS, NOT RADIANS/SECOND:
C  TAKEN FROM LEISSA, ''VIBRATION OF PLATES'', NASA SP-160, P. 44,
C  EQ. 4.20:
C
      FREQ(ILOADX)
     1       = SQRT(D/DMASS)*((1./LENGTH)**2 + (1./WIDTH)**2)*.5*PI
C
   50 CONTINUE
```

(d) GENOPT-user-written code for insertion into SUBROUTINE BEHX4:

```
C
C   PURPOSE: OBTAIN Normal deflection under uniform pressure
C
C  AS WITH THE VIBRATION OF THE UNLOADED PLATE, HERE WE ARE
C  NOT CONCERNED WITH THE NORMAL DEFLECTION UNDER AXIAL COMPRESSION
C  (Nx, ILOADX=1) OR UNDER UNIFORM IN-PLANE SHEAR (Nxy, ILOADX=2).
C
       W(ILOADX) = 0
       GO TO (50,50,30),ILOADX
C
   30 CONTINUE
C
C  TAKEN FROM ROARK, 3RD EDITION, TABLE X, FORMULA 36, P. 203, 1954:
C  NOTE:  VALID ONLY FOR LINEAR THEORY!!!
C
      W(ILOADX) = .1422*PRESS(ILOADX)*WIDTH**4/
     1          (E*THICK**3*(1. +2.21*(WIDTH/LENGTH)**3))
C
   50 CONTINUE
```

(e) GENOPT-user-written code for insertion into SUBROUTINE OBJECT:

```
C
C   PURPOSE:Weight of the plate
C
C  CALCULATE THE WEIGHT OF THE RECTANGULAR PLATE...
C
      WEIGHT = RHO*LENGTH*WIDTH*THICK
C
```

NOTES:

(1) These five algorithms are to be inserted in positions analogous to those indicated in TABLE 5 for BEHX1, BEHX4, and OBJECT. (BEHX2 and BEHX3 are omitted in TABLE 5 to save space.)

(2) Notice in (b) that the buckling loads for axial compression and in-plane shear are calculated from formulas that include the use of tables of buckling coefficients v. the aspect ratio of the plate, LENGTH/WIDTH. The actual tabular values of the buckling load coefficients KAXL for axial buckling and KSHR for shear buckling will be provided later by the end user. The GENOPT user provided the OPPORTUNITY for the end user to compute the buckling load factors with the use of interpolated coefficients. This illustrates an important point: GENOPT can be used for setting up simple program systems for optimizing things for which there exist design curves that can be expressed in the computer in tabular form. The linear interpolator, INTERP, is located in the GENOPT system library UTIL.NEW.

GENOPT user which are to be inserted in each of the BEHXn, $n = 1, 2, 3, 4$, routines for prediction of the various behaviors of the plate which may constrain the design during optimization iterations. Table 6 also lists the simple algorithm created by the GENOPT user for the objective function, which is to be inserted in the skeletal subroutine OBJECT, also located in the BEHAVIOR.NEW library.

Generation of absolute elements. After the GENOPT user has written/gathered the subroutines and algorithms to calculate the types of behavior that he or she has identified in GENTEXT as possibly constraining the design, he or she generates all the program system absolute elements needed by the end user for optimizing specific items within the generic class. The GENOPT command "GENPROGRAMS" generates the absolute elements. Table 15 of Bushnell (1989) lists the results of a "GENPROGRAMS" session and Table 16 of Bushnell (1989) lists the files generated.

The GENOPT user has now completed his/her tasks. The end user next chooses a specific case within the generic class.

EXAMPLE 1: MINIMUM WEIGHT DESIGN OF A PLATE

Part 2: Tasks performed by the end user

The end user now performs the tasks listed in Section 2 of Table 1. In this case the end user chooses a specific case name "PLATE1". In order to save space only a small part of the output from this example will be listed here. The reader is referred to Bushnell (1989) for more details.

In the interactive "BEGIN" session the end user supplies starting dimensions THICK, LENGTH, WIDTH; material properties E, NU, RHO; tables of buckling coefficients KAXL for pure axial compression and KSHR for pure in-plane shear; loads Nx, Ny, Nxy and p for each of three load sets; and allowables and factors of safety for each of the three load sets for each of the four behaviors listed in Table 4. In the interactive "BEGIN" session the end user is prompted for input and helped by the text listed in Table 3. Part of an interactive "BEGIN" session is listed in Table 17 of Bushnell (1989). This interactive session creates an annotated file of the input data called PLATE1.BEG in this example. Part of PLATE1.BEG is listed in Table 7. Note that the phrases and data names that follow the dollar signs in Table 7 are the same as the one-line definitions and data names listed in Table 3 on lines with leading indices of the form xx.1. Recall that these one-line definitions and data names were created by the GENOPT user with end user-friendliness in mind.

The end user next starts the interactive "DECIDE" session. He or she chooses decision variables, lower and upper bounds of the decision variables, linked variables and the linking expressions (equality constraints), inequality expressions, and "escape" variables. The linking expressions and inequality expressions are in the form of power series of the decision variable candidates. Tables 19 and 20 in Bushnell (1989) list part of the interactive "DECIDE" session and the annotated input file PLATE1.DEC created by this session. The format of PLATE1.DEC is analogous to the format of PLATE1.BEG, part of which is listed here in Table 7.

Following execution of "DECIDE", the end user conducts a very brief interactive "MAINSETUP" session. The interactive input session and the resulting annotated file PLATE1.OPT are listed in Tables 21 and 22, respectively, of Bushnell (1989).

The end user next launches a batch run via the command "OPTIMIZE". This batch run creates several files with names PLATE1.*, in which "*" signifies "any alphanumeric string". The files called PLATE1.OPM and PLATE1.OPP are the most important. PLATE.OPP lists all decision variables and design margins and the value of the objective function since the beginning of the case (after the command "OPTIMIZE" has been given possibly many times). PLATE1.OPM lists the results from each design iteration. Table 8 contains an abridged version of PLATE1.OPM. Note that the output contains the names established by the GENOPT user. The meanings of these names can be looked up in a glossary produced by GENOPT and stored in a file called NAME.DEF

Table 7. Part of the file PLATE1.BEG generated by "BEGIN"

```
================================================================
    N          $ Do you want a tutorial session and tutorial output?
0.1000000      $ thickness of the plate: THICK
      10       $ Length of the plate: LENGTH
 6.666700      $ Width of the plate: WIDTH
0.1000000E+08  $ Young's modulus of the plate material: E
0.3000000      $ Poisson's ratio of the plate material: NU
0.1000000      $ Weight density (lb/in**3) of the plate material: RHO
      16       $ Number IAOBAX of rows in the array  AOBAXL: IAOBAX
0.2000000      $ (plate length, LENGTH)/(plate width, WIDTH): AOBAXL(1)
0.3000000      $ (plate length, LENGTH)/(plate width, WIDTH): AOBAXL(2)


          (MANY LINES SKIPPED IN ORDER TO SAVE SPACE.)


       0   $ Maximum allowable normal deflection under pressure: AW(1)
       0   $ Maximum allowable normal deflection under pressure: AW(2)
0.1000000  $ Maximum allowable normal deflection under pressure: AW(3)
       0   $ Factor of safety for max deflection under pressure: FW(1)
       0   $ Factor of safety for max deflection under pressure: FW(2)
       1   $ Factor of safety for max deflection under pressure: FW(3)
================================================================
NOTE:  In the last six entries the loop over 3 corresponds to looping
over the number of load cases.  The allowables AW(1) and AW(2) and
the factors of safety FW(1) and FW(2) are zero because there is no
normal deflection of the plate corresponding to load case 1 (pure axial
compression) nor to load case 2 (pure in-plane shear).
```

Table 8. Part of the file PLATE1.OPM generated during the batch run launched by the end user's command "OPTIMIZE"

```
================================================================
$ OPTIMIZE      (Launch batch run for first set of 5 design
                 iterations)

ENTER THE SPECIFIC CASE NAME:  PLATE1

(YOU HAVE JUST LAUNCHED A BATCH RUN ON THE FAST QUEUE. WHEN IT IS
COMPLETED, YOU INSPECT THE  PLATE1.OPM  FILE TO SEE WHAT TO DO NEXT.)


 STRUCTURAL ANALYSIS FOR DESIGN ITERATION NO.   0:


          SUMMARY OF INFORMATION FROM OPTIMIZATION ANALYSIS
VAR. DEC. ESCAPE LINK. LINKED LINKING   LOWER   CURRENT   UPPER  VARIABLE
 NO. VAR.  VAR.  VAR.   TO   CONSTANT   BOUND    VALUE    BOUND    NAME
  1   Y     Y     N      0   0.00E+00   0.03  1.0000E-01   1.00   THICK
  2   Y     N     N      0   0.00E+00   5.00  1.0000E+01 100.0    LENGTH
  3   Y     N     N      0   0.00E+00   5.00  6.6667E+00  10.0    WIDTH

 ***** RESULTS FOR LOAD SET NO.  1  ******
 THOSE MARGINS LESS THAN UNITY CORRESPONDING TO CURRENT DESIGN
 MAR.   CURRENT
 NO.     VALUE            DEFINITION
  1   -2.655E-01  BUCKLE(1 )/(MINBUC(1 ) X FBUCKL(1 )) -1; F.S.=  1.20
```

Table 8—*Continued*

```
================================================================
***** RESULTS FOR LOAD SET NO.  2  ******
THOSE MARGINS LESS THAN UNITY CORRESPONDING TO CURRENT DESIGN
MAR.   CURRENT
NO.      VALUE            DEFINITION
 1    4.973E-02  MAXSTR(2 )/(STRESS(2 ) X FSTRES(2 )) -1; F.S.=  1.10
 2   -1.978E-01  BUCKLE(2 )/(MINBUC(2 ) X FBUCKL(2 )) -1; F.S.=  1.20

***** RESULTS FOR LOAD SET NO.  3  ******
THOSE MARGINS LESS THAN UNITY CORRESPONDING TO CURRENT DESIGN
MAR.   CURRENT
NO.      VALUE            DEFINITION
 1    2.722E-01  MAXSTR(3 )/(STRESS(3 ) X FSTRES(3 )) -1; F.S.=  1.00
 2   -2.613E-01  FREQ(3 )/  (MINCPS(3 ) X FSFREQ(3 )) -1; F.S.=  1.00
 3   -5.091E-01  AW(3 )/(W(3 ) X FW(3 )) -1;              F.S.=  1.00

******************** DESIGN OBJECTIVE ********************
   CURRENT VALUE OF THE OBJECTIVE FUNCTION:
VAR.   CURRENT
NO.      VALUE            DEFINITION
 1    6.667E-01  Weight of the plate: WEIGHT
************************************************************


   (MANY LINES SKIPPED IN ORDER TO SAVE SPACE.  ONLY THE WEIGHT OF
    THE PLATE FOR EACH DESIGN ITERATIONS IS GIVEN FOR THE NEXT FOUR
    ITERATIONS IN THIS PARTICULAR EXECUTION OF "OPTIMIZE".)


      5.940E-01  Weight of the plate: WEIGHT
      5.093E-01  Weight of the plate: WEIGHT
      4.902E-01  Weight of the plate: WEIGHT
      4.747E-01  Weight of the plate: WEIGHT


STRUCTURAL ANALYSIS FOR DESIGN ITERATION NO.   5:

          SUMMARY OF INFORMATION FROM OPTIMIZATION ANALYSIS
VAR. DEC. ESCAPE LINK. LINKED LINKING   LOWER   CURRENT   UPPER  VARIABLE
 NO. VAR.  VAR.  VAR.   TO   CONSTANT   BOUND    VALUE    BOUND    NAME
  1   Y     Y     N      0    0.00E+00   0.03  9.5263E-02   1.00   THICK
  2   Y     N     N      0    0.00E+00   5.00  1.0002E+01 100.0    LENGTH
  3   Y     N     N      0    0.00E+00   5.00  5.0000E+00  10.0    WIDTH

***** RESULTS FOR LOAD SET NO.  1  ******
THOSE MARGINS LESS THAN UNITY CORRESPONDING TO CURRENT DESIGN
MAR.   CURRENT
NO.      VALUE            DEFINITION
 1    2.328E-03  1.- (1.00-1.00*VAR(2)**(1.00)+50.0*VAR(3)**(-1.00))
 2    4.187E-02  BUCKLE(1 )/(MINBUC(1 ) X FBUCKL(1 )) -1; F.S.=  1.20

***** RESULTS FOR LOAD SET NO.  2  ******
THOSE MARGINS LESS THAN UNITY CORRESPONDING TO CURRENT DESIGN
MAR.   CURRENT
NO.      VALUE            DEFINITION
 1   -5.960E-08  MAXSTR(2 )/(STRESS(2 ) X FSTRES(2 )) -1; F.S.=  1.10
 2    1.463E-01  BUCKLE(2 )/(MINBUC(2 ) X FBUCKL(2 )) -1; F.S.=  1.20
```

Table 8—*Continued*

```
==========================================================================
***** RESULTS FOR LOAD SET NO.  3  ******
THOSE MARGINS LESS THAN UNITY CORRESPONDING TO CURRENT DESIGN
MAR.   CURRENT
NO.      VALUE            DEFINITION
 1    6.753E-01  MAXSTR(3 )/(STRESS(3 ) X FSTRES(3 )) -1; F.S.=  1.00
 2    5.657E-02  FREQ(3 )/  (MINCPS(3 ) X FSFREQ(3 )) -1; F.S.=  1.00
 3    3.438E-02  AW(3 )/(W(3 ) X FW(3 )) -1;              F.S.=  1.00

******************** DESIGN OBJECTIVE ********************
   CURRENT VALUE OF THE OBJECTIVE FUNCTION:
VAR.   CURRENT
NO.      VALUE            DEFINITION
 1    4.764E-01  Weight of the plate: WEIGHT
**********************************************************


 ADS DID NOT CHANGE THE DESIGN   (Because the design converged to an
                                  optimum!)


***** RESULTS FOR LOAD SET NO.  1  ******
PARAMETERS WHICH DESCRIBE BEHAVIOR (e.g. stress, buckling load)
BEH.   CURRENT
NO.      VALUE            DEFINITION
 1    1.050E+04    Maximum effective (von Mises) stress: STRESS(1 )
 2    1.250E+00    Buckling load factor: BUCKLE(1 )
 3    1.000E+10    Fundamental frequency of unloaded plate: FREQ(1 )
 4    1.000E-10    Normal deflection under uniform pressure: W(1 )

***** RESULTS FOR LOAD SET NO.  1  ******
MARGINS CORRESPONDING TO CURRENT DESIGN (F.S.= FACTOR OF SAFETY)
MAR.   CURRENT
NO.      VALUE            DEFINITION
 1    9.998E+00  1.00-1.00*VAR(2)**(1.00)+100.0*VAR(3)**(-1.00) -1.
 2    2.328E-03  1.- (1.00-1.00*VAR(2)**(1.00)+50.0*VAR(3)**(-1.00))
 3    5.002E+00  1.- (1.00-1.00*VAR(2)**(1.00)+1.00*VAR(3)**(1.00))
 4    1.598E+00  MAXSTR(1 )/(STRESS(1 ) X FSTRES(1 )) -1; F.S.=  1.10
 5    4.187E-02  BUCKLE(1 )/(MINBUC(1 ) X FBUCKL(1 )) -1; F.S.=  1.20

***** RESULTS FOR LOAD SET NO.  2  ******
PARAMETERS WHICH DESCRIBE BEHAVIOR (e.g. stress, buckling load)
BEH.   CURRENT
NO.      VALUE            DEFINITION
 1    2.727E+04    Maximum effective (von Mises) stress: STRESS(2 )
 2    1.376E+00    Buckling load factor: BUCKLE(2 )
 3    1.000E+10    Fundamental frequency of unloaded plate: FREQ(2 )
 4    1.000E-10    Normal deflection under uniform pressure: W(2 )

***** RESULTS FOR LOAD SET NO.  2  ******
MARGINS CORRESPONDING TO CURRENT DESIGN (F.S.= FACTOR OF SAFETY)
MAR.   CURRENT
NO.      VALUE            DEFINITION
 1   -5.960E-08  MAXSTR(2 )/(STRESS(2 ) X FSTRES(2 )) -1; F.S.=  1.10
 2    1.463E-01  BUCKLE(2 )/(MINBUC(2 ) X FBUCKL(2 )) -1; F.S.=  1.20
```

Table 8—*Continued*

```
==================================================================
***** RESULTS FOR LOAD SET NO.  3  ******
PARAMETERS WHICH DESCRIBE BEHAVIOR (e.g. stress, buckling load)

BEH.   CURRENT
NO.     VALUE            DEFINITION
 1    1.791E+04    Maximum effective (von Mises) stress: STRESS(3 )
 2    1.000E+10    Buckling load factor: BUCKLE(3 )
 3    1.374E+02    Fundamental frequency of unloaded plate: FREQ(3 )
 4    9.668E-02    Normal deflection under uniform pressure: W(3 )

***** RESULTS FOR LOAD SET NO.  3  ******
MARGINS CORRESPONDING TO CURRENT DESIGN (F.S.= FACTOR OF SAFETY)
MAR.   CURRENT
NO.     VALUE            DEFINITION
 1    6.753E-01  MAXSTR(3 )/(STRESS(3 ) X FSTRES(3 )) -1; F.S.=  1.00
 2    5.657E-02  FREQ(3 )/  (MINCPS(3 ) X FSFREQ(3 )) -1; F.S.=  1.00
 3    3.438E-02  AW(3 )/(W(3 ) X FW(3 )) -1;              F.S.=  1.00
**********  ALL 3 LOAD CASES PROCESSED **********
**************************************************
==================================================================
```

(NAME = "PLATE" in this example). For each datum this glossary lists:

(1) whether or not the datum is an array;
(2) the number of rows and columns in the array;
(3) the role of the datum (1–7, as listed in Section 1 of Table 1 and for this example in Table 2);
(4) the index number of the prompt for the datum (numbers preceding the decimal point on the left side of Table 3);
(5) the name of the datum;
(6) the GENOPT-user supplied one-line definition of the datum.

Table 7 of Bushnell (1989) lists the glossary for this example. (Table 29 in Bushnell (1989) lists the complete glossary for the much more complex Example 2, to be discussed below.)

Note from Table 8 that convergence to an optimum design is obtained in this simple case after only five design iterations. This is unusual. With more practical cases, such as the rather large case described in the next section, the command "OPTIMIZE" must be given several times before convergence is obtained. Each time the command "OPTIMIZE" is given the starting design is the same as the final design after the previous "OPTIMIZE" run.

As seen from Table 8, at the converged optimum design, there are five margins that are very small, two from Load Set 1, one from Load Set 2, and two from Load Set 3. These critical margins have obviously acted as constraints on the design as it evolved during design iterations.

Figures 3–5 are GENOPT-generated plots of the history of the design process. The end user obtains plots such as these via the interactive "CHOOSEPLOT" and "DIPLOT" processors, both of which are described in Bushnell (1989).

The files generated by actions of the end user are described in an interactive HELP utility which is activated by the user's typing the command "HELPG". Further details are given in Bushnell (1989).

EXAMPLE 2: MINIMUM WEIGHT DESIGN OF A COMPOSITE FLAT OR CURVED PANEL OR CYLINDRICAL SHELL WITH COMPOSITE STRINGERS AND/OR RINGS

Part 1: *Tasks performed by the GENOPT user*

Unlike Example 1, which is a bit contrived, this example represents a practical problem.

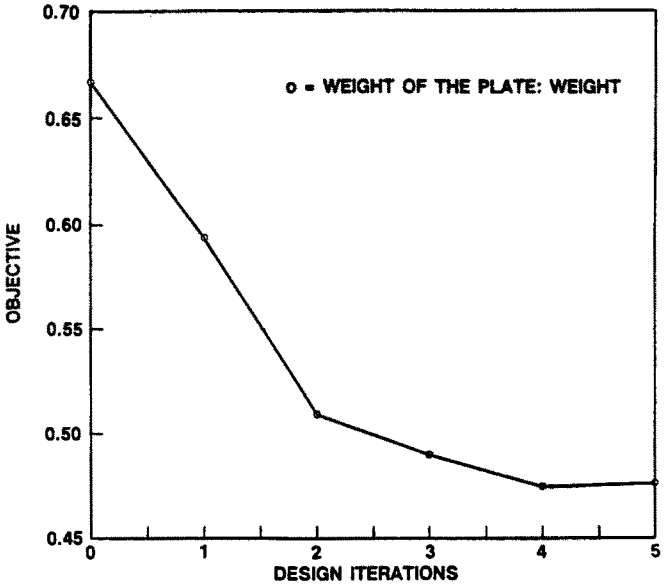

Fig. 3. Example 1: minimum weight design of rectangular, isotropic plate vs design iterations. A maximum of 10 iterations was allowed, but convergence of the design to an optimum was achieved after only five iterations. The number of iterations required for convergence is usually much greater.

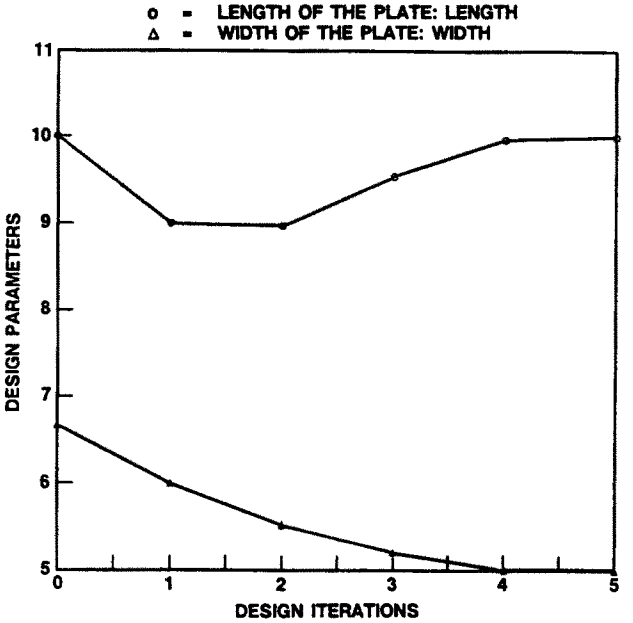

Fig. 4. Example 1: minimum weight design of plate. Decision variables LENGTH and WIDTH vs design iterations.

The purpose here is to determine what problems might be encountered in forming the absolute element "OPTIMIZE" via the following process:

(1) "borrowing" and modifying previously developed software from another source.
(2) creating a new library of subroutines called ADDCODE1.NEW,
(3) inserting labelled common blocks and subroutine calls at the appropriate locations in SUBROUTINE STRUCT, and
(4) collecting FORTRAN object elements for code that requires no changes for use in the GENOPT system.

In this example the absolute element "OPTIMIZE" permits minimum weight design of rather complex, composite, curved, stiffened panels subject to multiple sets of combined in-plane loads. The "borrowed" subroutines are from PANDA2 (Bushnell, 1987). The

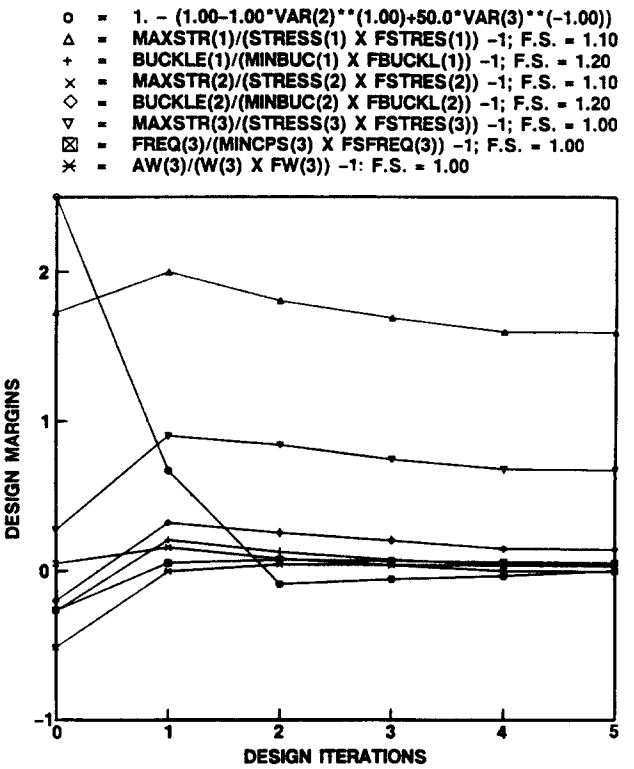

Fig. 5. Example 1: minimum weight design of plate. History of design margins vs design iterations. The quantities in the legend are defined as follows: VAR(2), VAR(3) = LENGTH, WIDTH (see Table 8); the remaining variables (MAXSTR, STRESS, FSTRESS, BUCKLE, MINBUC, FBUCKL, etc.) are defined in Table 2. F.S. = "factor of safety".

theoretical basis of the analysis is described in Bushnell (1987) and Bushnell (1983b). More details on this example are provided in the files GENOPTST.ORY and PANEL.CAS in the GENOPT program system (Bushnell, 1989).

Statement of the problem. The composite, curved or flat, stiffened panels can be subjected to up to 20 combinations of uniform in-plane stress resultants, Nx, Ny, Nxy (axial, transverse or "hoop", in-plane shear, respectively). The multiple combinations might represent environments seen by the structure during different phases of its operation. The composite panel may be made of several different materials. The panel may be unstiffened or reinforced by stringers or rings or both.

In this example there are, for each load combination, 18 types of behavior which might constrain the design. Most of these are types of buckling. The various kinds of buckling are listed in Table 1 of Bushnell (1983b). Types of buckling accounted for are:

(1) general instability,
(2) buckling between rings with "smeared" stringers (Baruch and Singer, 1963), curvature included,
(3) buckling between rings with "smeared" stringers, wide column model,
(4) local buckling between adjacent rings and stringers,
(5) local buckling of the various segments (webs, flanges) of both stringers and rings,
(6) various "rolling" or "tripping" modes of the stiffeners, both including and neglecting participation of the panel skin between the stiffeners, and
(7) an axisymmetric "rolling" of rings on complete cylindrical shells.

In addition to this variety of buckling phenomena, five types of stress are computed in material coordinates (referring to the fiber direction) of each lamina of each segment of the composite structure:

(1) maximum tensile stress along the fibers,

(2) maximum compressive stress along the fibers,
(3) maximum tensile stress normal to the fibers,
(4) maximum compressive stress normal to the fibers,
(5) maximum in-plane shear stress.

Behavioral constraints are introduced for the maximum stress in each category for each type of material used in the panel.

Preparation for the "GENTEXT" interactive session. (1) Finding models for predicting behavior of the stiffened panels:

The first task of the GENOPT user is to assemble the various algorithms needed to predict the behaviors just listed. In this case all of the algorithms were "borrowed" from the PANDA2 program system (Bushnell, 1987).

In PANDA2 a stiffened panel is considered to be an assemblage of panel modules as illustrated in Fig. 6a. Each module is considered to consist of several segments (four in Fig. 6b), each segment of which has its own laminated wall construction (Fig. 6c). Both cross-sections of panel skin and stringers and panel skin and rings are modelled in this way. Figure 6a shows a panel with three modules; Fig. 6b shows a single module with four segments; and Fig. 6c shows the layer numbering convention within each segment. Stiffeners with T. J. Rectangular (blade), of Hat cross-sections are permitted.

In this case it was necessary to modify some of the "borrowed" software because, as originally written, it contained calculation of behavioral constraints, whereas in GENOPT the actual constraint conditions are calculated in a GENTEXT-generated portion of STRUCT: the part with the calls of BEHX1, BEHX2,... BEHX*n* and CONX (Table 4).

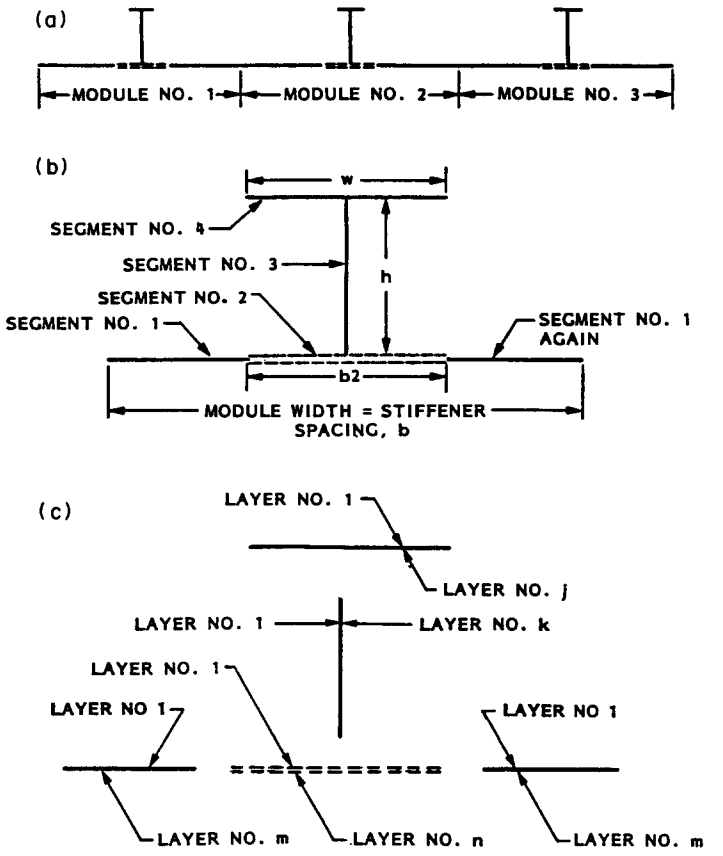

Fig. 6. Example 2: minimum weight design of stiffened composite panels. Approach created by the GENOPT user: (a) panel with T-shaped stringers, (b) a single module of the panel with segment numbering convention, (c) layer numbering convention.

The way in which stress constraints are calculated in PANDA2 had to be changed: in PANDA2 behavioral constraints are calculated corresponding to maximum stress components for each segment for each material type, whereas in this GENOPT application it was decided that only the overall maximum stress components for each material type be calculated in order not to require the introduction of too many allowables and factors of safety.

In this complex case, in which much software is "borrowed" from another source, the many different kinds of behavior (18 for each load set!) were calculated in subroutines the calls for which were introduced by the GENOPT user into SUBROUTINE STRUCT rather than into each of 18 BEHX*n* subroutines. A subroutine called TRANFR was written to translate data names from those established by the GENOPT user to those used by the developer of the "borrowed" code. Further details are provided in Bushnell (1989).

In this example the presence of many labelled common blocks in PANDA2 and the fact that PANDA2 already is an optimization code and not simply an analysis code made the process of creating a GENOPT-based panel optimization program more onerous than would be expected during generation of other GENOPT-based optimization programs. Analysis codes that do not contain too many labelled common blocks that must be included in STRUCT.NEW would be easier to insert and use with the GENOPT system.

As with Example 1, the objective is minimum weight of the panel.

In order to save space, the interactive GENTEXT session and the GENPROGRAMS session will not be shown here. They proceed in ways analogous to those shown in Example 1.

(2) Organizing and deciding on names and definitions for the input data:

In this complex example there are many more names, arrays, and pointers than was the case in the relatively simple Example 1. Therefore, the author does not introduce a table analogous to Table 2, but reproduces here as Table 9 part of the glossary that GENTEXT

Table 9. Abridged glossary of variables used in the generic case "PANEL" (Example 2). (Because of space limitations only those variables that appear in Figs 13 and 14 are included in this list. The complete glossary can be found in Bushnell (1989).

ARRAY ?	NUMBER OF (ROWS,COLS)	ROLE	PROMPT NUMBER (PANEL.PRO)	VAR. NAME	DEFINITION OF THE VARIABLE
y	(20, 0)	4	280	GENBUC	= General instability load factor
y	(20, 0)	5	285	AGENBK	= Allowable for general instability load factor (use 1.0)
y	(20, 0)	6	290	FSGEN	= Factor of safety for general instability load factor
y	(20, 0)	4	295	WIDCOL	= Wide column buckling between rings load factor
y	(20, 0)	5	300	AWIDCL	= Allowable for wide column buckling load factor (use 1.0)
y	(20, 0)	6	305	FSWID	= Factor of safety for wide column buckling load factor
y	(20, 0)	4	310	PANBUC	= Panel buckling load factor
y	(20, 0)	5	315	APANBK	= Allowable for panel buckling load factor (use 1.0)
y	(20, 0)	6	320	FSPAN	= Factor of safety for panel buckling load factor
y	(20, 0)	4	325	LOCBUC	= Local buckling load factor (panel skin)
y	(20, 0)	5	330	ALOCBK	= Allowable for panel skin buckling load factor (use 1.0)
y	(20, 0)	6	335	FSLOC	= Factor of safety for panel skin buckling load factor

Table 9—*Continued*

ARRAY ?	NUMBER OF (ROWS,COLS)	ROLE	PROMPT NUMBER (PANEL.PRO)	VAR. NAME	DEFINITION OF THE VARIABLE
y	(20, 0)	4	340	ROLSKN	= Local buckling load factor with stiffener rolling
y	(20, 0)	5	345	AROLSK	= Allowable local buck. with stiffener rolling (use 1.0)
y	(20, 0)	6	350	FROLSK	= Factor of safety for skin buckling with stiffener rolling
y	(20, 2)	4	435	ROLSTF	= Rolling of stiffeners without skin participation
y	(20, 2)	5	440	AROLST	= Allowable for rolling of stiffener without skin (use 1.0)
y	(20, 2)	6	445	FSROL1	= Factor of safety for rolling of stiffener, no skin motion
y	(20, 2)	4	450	ROLSMR	= Rolling of stiffeners with other set smeared
y	(20, 2)	5	455	AROLSM	= Allowable for rolling of stiffeners, other set smeared (use 1.0)
y	(20, 2)	6	460	FSROL2	= Factor of safety for rolling with other set smeared
n	(0, 0)	2	485	JSIG1T	= Index for type of material in SIG1T(NCASES,JSIG1T)
y	(20, 5)	4	490	SIG1T	= Max. tension along fibers for matl type
y	(20, 5)	5	495	ASIG1T	= Allowable stress for tension along fibers
y	(20, 5)	6	500	FSS1T	= Factor of safety for tension along fibers
y	(20, 5)	4	505	SIG1C	= Max. compression along fiber
y	(20, 5)	5	510	ASIG1C	= Allowable compression along fiber
y	(20, 5)	6	515	FSS1C	= Factor of safety for compression along fibers
y	(20, 5)	4	520	SIG2T	= Max. tensile stress normal to fibers
y	(20, 5)	5	525	ASIG2T	= Allowable tensile stress normal to fibers
y	(20, 5)	6	530	FSS2T	= Factor of safety for tensile stress normal to fibers
y	(20, 5)	4	535	SIG2C	= Max. compressive stress normal to fibers
y	(20, 5)	5	540	ASIG2C	= Allowable compressive stress normal to fibers
y	(20, 5)	6	545	FSS2C	= Factor of safety for compressive stress normal to fibers
y	(20, 5)	4	550	SIG12	= Max. in-plane shear stress
y	(20, 5)	5	555	ASIG12	= Allowable in-plane shear stress
y	(20, 5)	6	560	FSS12	= Factor of safety for in-plane shear stress
n	(0, 0)	7	570	WEIGHT	= Weight of the panel

NOTES:

(1) GENBUC...Number of rows = 20, corresponding to 20 load sets.
(2) SIG1T....Max. of 5 columns corresponding to 5 different materials.
(3) In order to save space, this table lists primarily the variables that appear in Figs. 13 and 14. The full glossary appears in [10].

produces in the file NAME.DEF. (In this case NAME = "PANEL"). The complete GENTEXT-generated glossary is given in Table 29 of Bushnell (1989). The generic case name is PANEL. Table 30 of Bushnell (1989) lists a small part of the input data prompting file, PANEL.PRO, in which the text was composed interactively by the GENOPT user during the GENTEXT session.

EXAMPLE 2: MINIMUM WEIGHT DESIGN OF A COMPOSITE FLAT OR CURVED PANEL OR CYLINDRICAL SHELL WITH COMPOSITE STRINGERS AND/OR RINGS

Part 2: Tasks performed by the end user

The end user now performs the tasks listed in Section 2 of Table 1. The runstream actually used by the end user in this case is listed in Table 10.

In this case the end user chooses a specific case name "ARIANE". A minimum weight design is to be obtained for the ARIANE4 interstage between the second and third stages of ARIANE4 booster. Blaas and Wiggenraad call it "Interstage 2.3" (Blass and Wiggenraad, 1986). Many details of the modelling philosophy and strategy are given in Bushnell (1987). Figures 7–9, taken from Bushnell (1987) show the structure, which is a complete cylindrical

Table 10. End user portion of the PANEL/ARIANE example

```
$ BEGIN        (You or your appointed user start a specific case.
                You call this case  ARIANE.  The BEGIN processor
                allows you to provide a starting design, material
                properties, loading, allowables, and factors of
                safety for multiple (in this case 2) load sets.)

$ DECIDE       (You choose decision variables, their lower and
                upper bounds; linked variables and their linking
                constants; and escape variables.)

$ MAINSETUP    (You choose what type of analysis to perform, and
                how many design iterations.)

$ OPTIMIZE     (You launch a batch run for the first 5 iterations.)
$ OPTIMIZE     (You launch a batch run for the second 5 iterations.)
$ OPTIMIZE     (You launch a batch run for the third 5 iterations.)
$ OPTIMIZE     (You launch a batch run for the fourth 5 iterations.)

$ CHANGE       (You interactively set some of the layer thicknesses
                equal to zero. They were getting very small.)

$ OPTIMIZE     (You launch a batch run for the fifth 5 iterations.)
$ OPTIMIZE     (You launch a batch run for the sixth 5 iterations.)
$ OPTIMIZE     (You launch a batch run for the seventh 5 iterations.)
$ OPTIMIZE     (You launch a batch run for the eighth 5 iterations.)

$ MAINSETUP    (You change number of design iterations from 5 to 10)

$ OPTIMIZE     (You launch a batch run for the ninth set of
                iterations.)

$ CHOOSEPLOT   (You choose several variables to be plotted v.
                iterations.)
$ DIPLOT       (You get plots from the laser printer.)
$ CHOOSEPLOT   (You choose several more variables to be plotted v.
                iterations.)
$ DIPLOT       (You get new plots from the laser printer.)
```

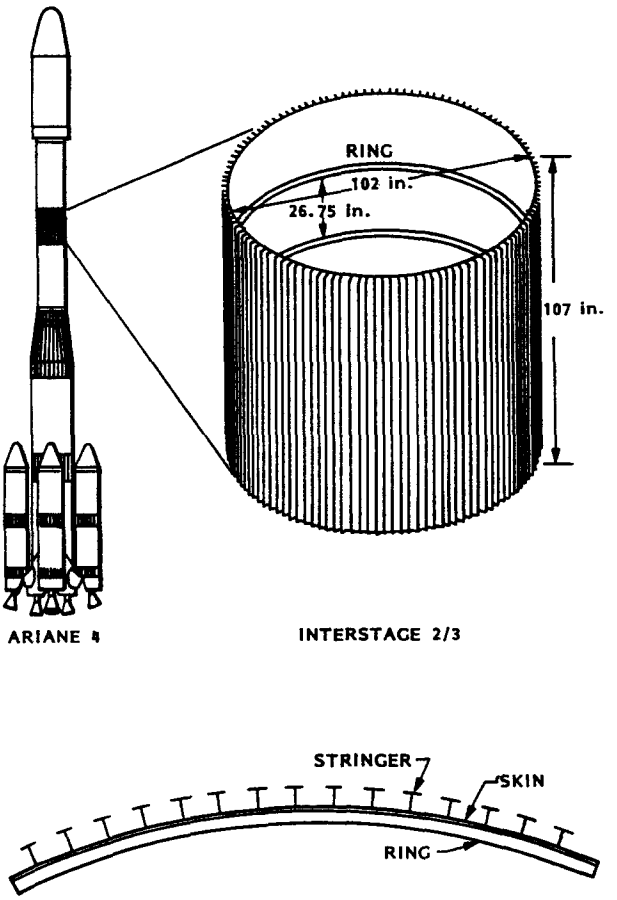

Fig. 7. Example 2: minimum weight design of stiffened composite panels. Specific case run by end user: ARIANE4 interstage between stage 2 and stage 3. The cylindrical shell is reinforced by external stringers and internal rings.

shell with external T-shaped stringers and internal blade-shaped rings (Fig. 7); how the complete cylindrical shell with variable loads around its circumference is modelled as a single cylindrical panel with three different uniform loads (Fig. 8); and how a panel skin-stringer module is to be constructed (Fig. 9).

As explained in more detail in Bushnell (1987), the complete cylindrical shell under loads that vary around its circumference, is modelled as a single cylindrical panel spanning 40 inches of circumference and subjected to two sets of uniform in-plane loads: (1) pure axial compression of 3000 lb in.$^{-1}$ and (2) combined axial compression of 1000 lb in.$^{-1}$ and in-plane shear of 1000 lb in.$^{-1}$. It was found in Bushnell (1987) that the third load set, pure axial tension of 2000 lb in.$^{-1}$ does not generate any active behavioral constraint conditions during the optimization process.

The end user's runstream for the optimization is listed in Table 10. In order to save space, no details will be given here of the interactive "BEGIN", "DECIDE", "MAIN-SETUP", and "CHANGE" sessions. These and other details appear in the file PANEL.CAS in Bushnell (1989).

The results from "CHOOSEPLOT/DIPLOT" are shown in Figs 10–14. During the first 23 design iterations some of the wall thicknesses became very small. These were set equal to zero via an interactive "CHANGE" session (see Fig. 12). The evolution of the panel weight and of the design are shown in Figs 10–12. The variables are defined in the legend. Notice that variables of like magnitude are plotted together.

Figure 13 shows the evolution of the more critical design margins corresponding to the first load set ($Nx = -3000$ lb in.$^{-1}$); Fig. 14 shows the evolution of the more

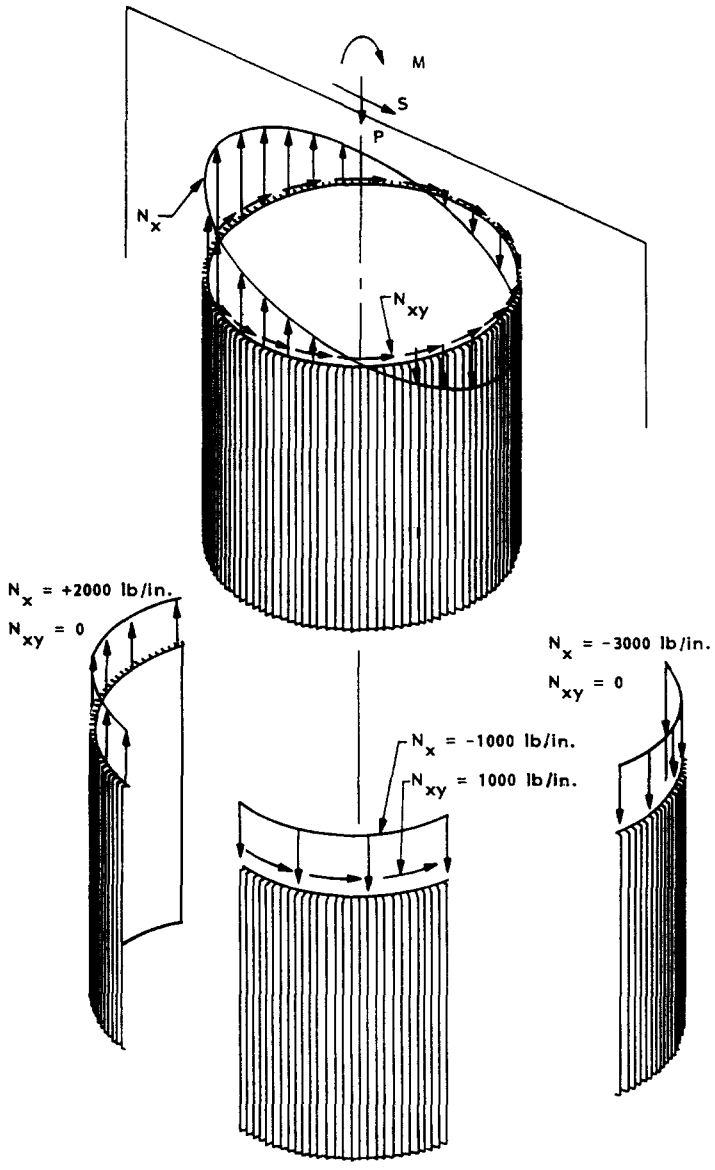

Fig. 8. Example 2: minimum weight design of stiffened composite panels. Specific case run by end user: applied moment M, axial compression P and shear S; assumed resulting distributions of line loads Nx and Nxy in the shell; and replacement of the actual complete cylindrical shell with circumferentially varying line loads by a single cylindrical panel spanning 40 in. of circumference with three separate load sets, each of which has uniform line loads Nx and Nxy. [In Example 2 the load case with $Nxy = 0$, $Nx = +2000$ lb in.$^{-1}$ is not used because a previous study (Bushnell, 1987) found that this load set created no critical behavioral constraints.]

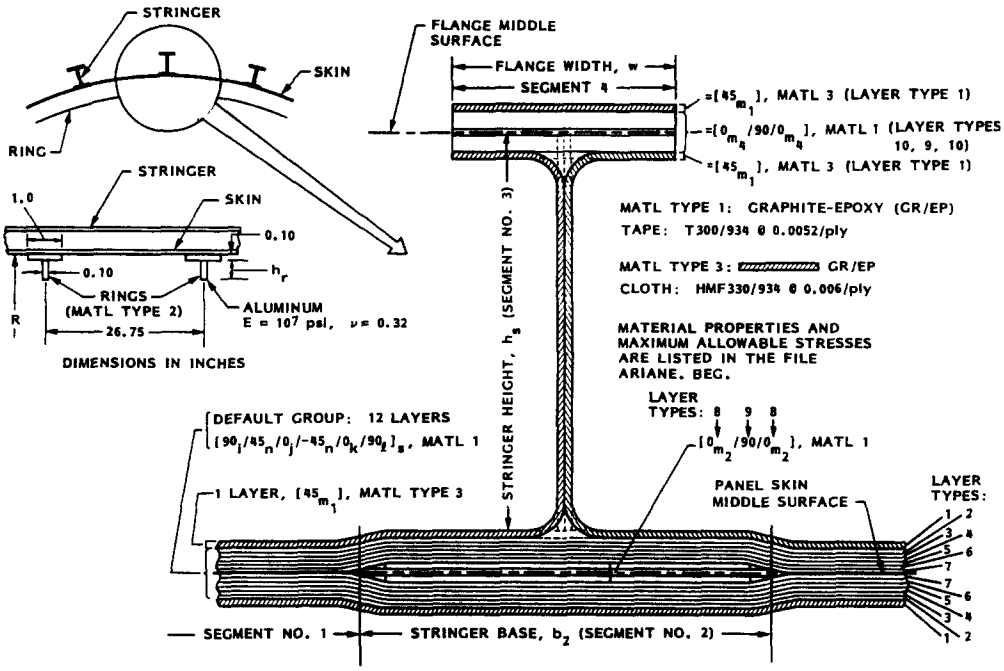

Fig. 9. Example 2: minimum weight design of stiffened composite panels. Specific case run by end user: design concept to be optimized: external T-shaped stringers made of graphite-epoxy cloth and tape with thickened areas under the stringer web, and internal aluminum rings with spacing fixed at 26.75 in. The spacing of the stringers is to be determined, as well as the cross-section dimensions and thicknesses of the various composite laminae.

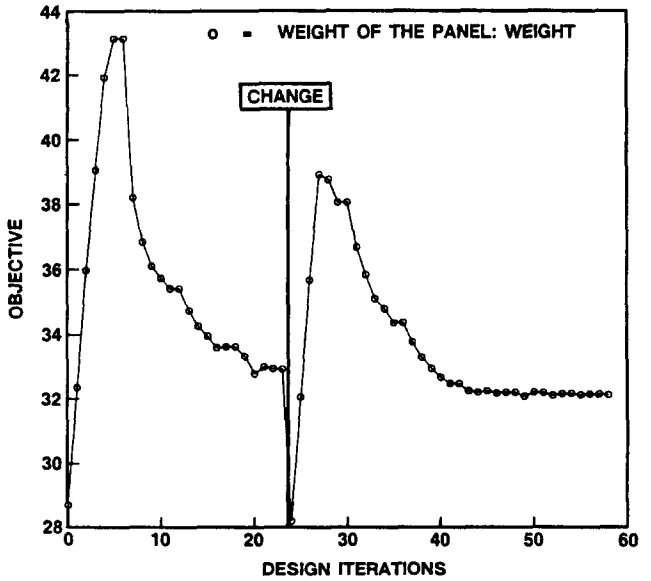

Fig. 10. Example 2: minimum weight design of stiffened composite panels. Weight vs design iterations for all nine "OPTIMIZES" listed in Table 10. The "CHANGE" processor was used after four "OPTIMIZES" in order to set several of the lamina thicknesses equal to zero. (See Fig. 12.)

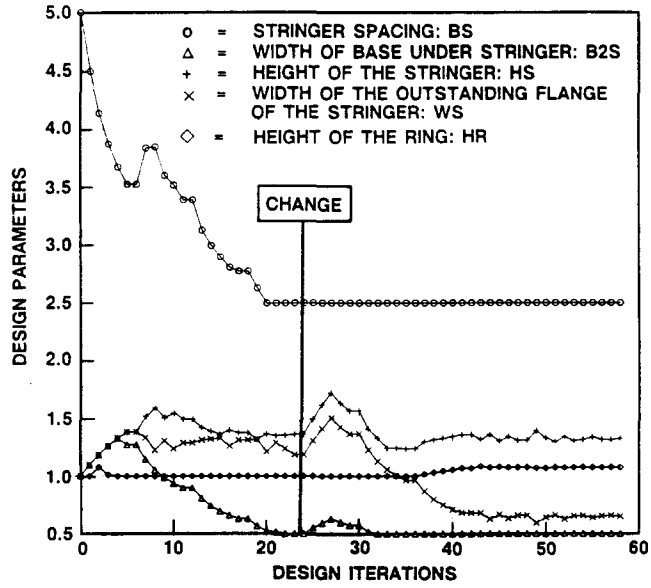

Fig. 11. Example 2: minimum weight design of stiffened composite panels. Cross section dimensions vs design iterations.

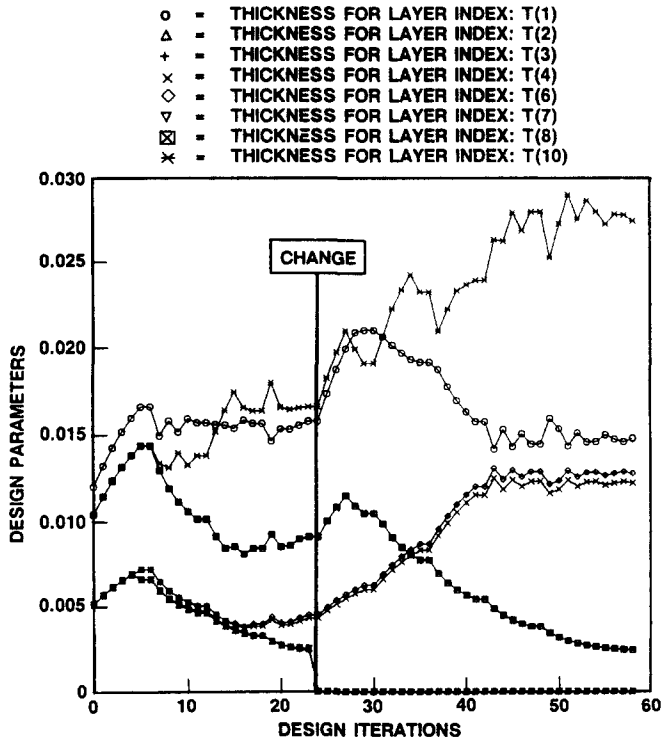

Fig. 12. Example 2: minimum weight design of stiffened composite panels. Laminae thicknesses (see Fig. 9) vs design iterations. The thickness for Layer Index 5 is the same as that for Layer Index 3 and the thickness for Layer Index 9 is fixed at 0.0052 in.

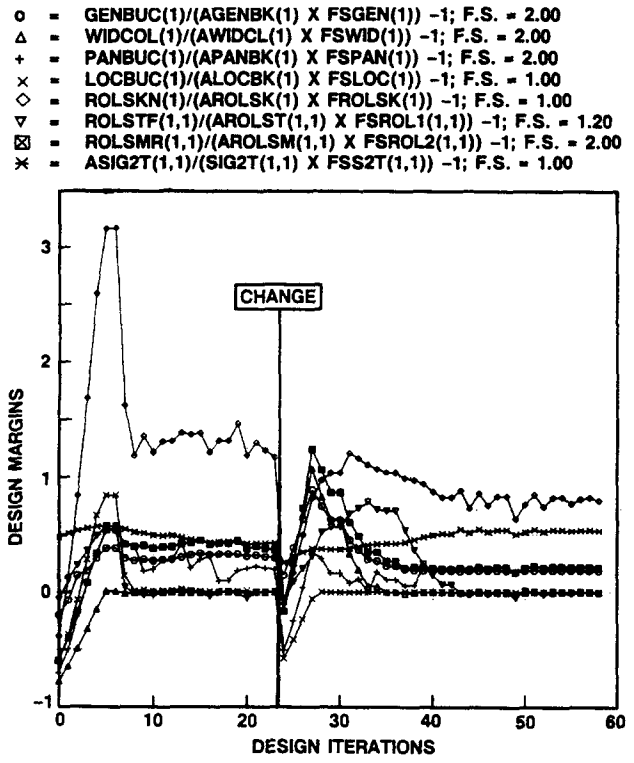

Fig. 13. Example 2: minimum weight design of stiffened composite panels. Significant margins vs design iterations for the first load set (pure axial compression, $Nx = -3000$ lb in.$^{-1}$). The variables (GENBUC, AGENBK, etc.) in the legend are defined in Table 9. F.S. = "factor of safety".

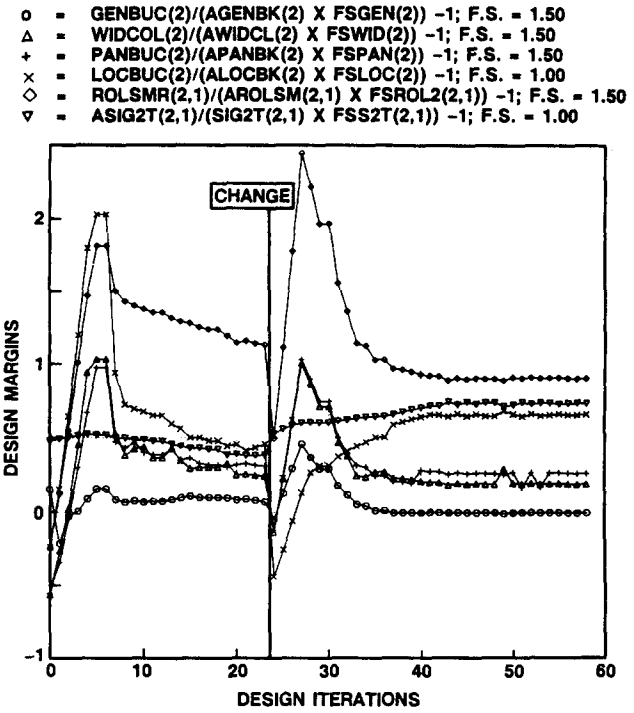

Fig. 14. Example 2: minimum weight design of stiffened composite panels. Significant margins vs design iterations for the second load set (axial compression, $Nx = -1000$ lb in.$^{-1}$, in-plane shear, $Nxy = 1000$ lb in.$^{-1}$). The variables in the legend are defined in Table 9. F.S. = "factor of safety".

critical design margins corresponding to the second load set ($Nx = -1000$ lb in^{-1}., $Nxy = 1000$ lb in.$^{-1}$). Again, the variables plotted are defined in the legend.

Acknowledgements—This work was sponsored by the 1987–1988 Lockheed Independent Development Program. The author is indeed grateful for the continuing support of Mr Bill Sable, Structural Analysis and Test Department in Lockheed Missiles and Space Company's Satellite Systems Division. Ms Karen Neier of Department 97-10 of the Lockheed Palo Alto Research Laboratories wrote the original DIPLOT package which was modified for use with GENOPT. Harold Cabiness, Chris Dumlao and Jörgen Skogh of the Computational Mechanics Section of Department 93-30 of the Lockheed Palo Alto Research Laboratories reviewed this paper, tested GENOPT, and offered many suggestions for improvements. The author is especially grateful for the many tests of GENOPT performed by Harold Cabiness and for his insistence that GENOPT be made more user-friendly and that unnecessary pitfalls be eliminated.

REFERENCES

Baruch, M. and Singer, J. (1963). Effect of eccentricity of stringers on the general instability of stiffened cylindrical shells under hydrostatic pressure. *J. Mech. Engng Sci.* **5**, 23–27.

Blass, C. and Wiggenraad, J. F. M. (1986). Development and test verification of the ARIANE4 interstage 2.3 in CFRP, *Proc. AIAA/ASME 27th Structures, Structural Dynamics Materials Conf., Part* 1, pp. 307–313.

Bushnell, D. (1983a). ACTUATOR—an interactive computer program for optimum design of mirror/actuator systems. (Unpublished paper.)

Bushnell, D. (1983b). PANDA—interactive program for minimum weight design of stiffened cylindrical panels and shells. *Comput. Struct.* **16**, 167–185.

Bushnell, D. (1986a). BOSOR4: Program for stress, stability, and vibration of complex, branched shells of revolution. In *Structural Analysis Systems* (Edited by A. Niku-Lari), Vol. 2, pp. 25–54. Pergamon, Oxford.

Bushnell, D. (1986b). BOSOR5: Program for buckling of complex, branched shells of revolution including large deflections, plasticity and creep. In *Structural Analysis Systems* (Edited by A. Niku-Lari), Vol. 2, pp. 55–67. Pergamon, Oxford.

Bushnell, D. (1987). Theoretical basis of the PANDA computer program for preliminary design of stiffened panels under combined in plane loads. *Comput. Struct.* **27**, 541–563.

Bushnell, D. (1987). PANDA2—program for minimum weight design of stiffened, composite, locally buckled panels. *Comput. Struct.* **25**, 469–605.

Bushnell, D. (1988). Improved optimum design of dewar supports. *Comput. Struct.* **29**, 1–56.

Bushnell, D. (1989). The files called GENOPTST.ORY, GENOPT.HLP, PLATE.*, PLATE1.*, PANEL.*, ARIANE.* and HOWTO.RUN included with other files in the GENOPT program system. (Unpublished.)

Leissa, A. (1969). Vibration of plates. NASA SP-160, p. 44, Eqn 4.20. NASA Langley Research Ctr, Hampton, VA.

Roark, R. J. (1954) *Formulas for Stress and Strain*, 3rd edn. McGraw-Hill, New York.

Vanderplaats, G. N. (1987). ADS—a FORTRAN program for automated design synthesis, Version 2.01, Engineering Design Optimization, Santa Barbara, CA. (Unpublished user's manual.)

Vanderplaats, G. N. and Sugiomoto, H. (1986). A general-purpose optimization program for engineering design, *Comput. Struct.* **24**, 13–21.