Role of Data Base Management in Design Optimization Systems

G. J. Park* and J. S. Arora†
University of Iowa, Iowa City, Iowa

To study the role of data base and data base management system, an interactive design optimization software system called IDESIGN5 is developed to solve nonlinear programming problems. Four promising algorithms are included to overcome the lack of unanimous choice of an algorithm. Tuning parameters and procedures of algorithms are implemented through extensive numerical experimentation. The interactive process consists of menu displays, advice for decisions, and applicable messages. Input data can be created interactively and the designer can change problem parameters, algorithm, and design variable data at any point of execution. If a design variable does not effect the optimization process, it can be given a fixed value interactively. Discrete variable optimization can be performed by using design variable status capability of the system. Graphics facilities are provided for decision making. The system consists of several modules that communicate with each other through a data base managed by a data base management system. Several example problems are solved in batch and interactive environments to test the system.

Introduction

THE nonlinear programming problem (NLP) in the finite-dimensional space is to minimize a cost function $f(b)$ for $b \in S$, where S is a subset of R^n defined as

$$S=\{b{:}g_i(b)=0;i=1,m';g_i(b)\leq 0;i=m'+1,m;$$
$$b_{iL}\leq b_i\leq b_{iU};i=1,n\}$$

and $g_i(b)$ are constraints, b a design variable vector, and b_{iL} and b_{iU} the smallest and largest values allowed for the ith design variable, respectively.

The problem has been treated in the nonlinear programming literature quite extensively. In that material, it is usually assumed that the explicit form of the function $f(b)$ and $g_i(b)$ is available. However, if we allow the functions to be implicit, then several classes of structural and mechanical system design problems can be described by the model.[1]

A number of nonlinear programming algorithms have been developed and used for optimal design.[2-6] However, software development for design optimization is lagging behind these advances.[7-11] It is well known that some algorithms when implemented in a computer program do not behave in the way they are theoretically supposed to. Considerable expertise and numerical experimentation is needed to properly implement an algorithm. In addition, no single algorithm can efficiently solve all classes of problems. Instead, a powerful software system with various capabilities and facilities is desired to minimize the difficulties. So far, software for design optimization is either not available or it is tedious to use. Most of the existing programs are not interactive. This means that the program does not allow any user control over the iterative process. Graphics capabilities is almost unheard of in design optimization. The software should have these facilities and options for various controls and decision making. Also, as range of applications of optimization expands, more complex data are generated which must be handled properly through a data base management system (DBMS). It should be possible to easily refine existing capabilities and add new ones. To allow various options, several good algorithms should be implemented.

To study the role of an organized data base and a data base management system, a general purpose software system called IDESIGN5 is introduced in this paper. It is an interactive design optimization system incorporating modern data base management concepts. The system has the foregoing capabilities and is an extension of an earlier program called IDESIGN3.[10] That program does not use any data base management system and has no out-of-core calculations. The new system is designed using modern software development and data base management concepts.

Selection of Algorithms for IDESIGN5

Although many algorithms are available for solving nonlinear optimization problems, it is difficult to find an algorithm that solves all classes of the problems efficiently. In IDESIGN5, several good algorithms have been selected. These are Pshenichny's algorithm (LINRM),[2] cost function bounding algorithm (CFB),[4] modified Pshenichny's algorithm (PLBA),[3] and a hybrid method.[6] These algorithms are selected based on their generality, robustness, and efficiency. They have performed reasonably well on a wide range of applications.

The detailed procedures of the selected algorithms have been published in the literature.[1-6] All algorithms roughly implement the following steps:

1) Set $k=0$; estimate starting design variables; initialize parameters.

2) At the kth iteration, identify a potential constraint set. Define and solve a subproblem for the search direction.

3) Check the convergence criteria. If the criteria are satisfied, the process is stopped.

4) Check the condition for the progress of the algorithm and calculate a step size, update the Hessian of Lagrangian if necessary.

5) Update the design variables. The process is then restarted from step 2.

Presented as Paper 86-0991 at the AIAA/ASME/ASCE/AHS 27th Structures, Structural Dynamics and Materials Conference, San Antonio, TX, May 21-23, 1986; received April 4, 1986; revision received March 15, 1987. Copyright © American Institute of Aeronautics and Astronautics, Inc., 1987. All rights reserved.

*Graduate Research Assistant, Optimal Design Laboratory, College of Engineering (currently Assistant Professor, CAD/CAM Center, Purdue University at Indianapolis, IN).

†Professor, Civil and Mechanical Engineering, Optimal Design Laboratory, College of Engineering. Member AIAA.

Role of Data Base in Design Optimization

IDESIGN5 implements the major steps of the selected algorithms in separate modules. A module is a program that can be compiled and executed independently. The sequence of execution of modules is controlled by the main module. The modules communicate with each other through the data base. Interaction is allowed during the execution of the system from the beginning to the end. The designer can control the progress by changing the problem data and algorithm at any point.

Interactive Capabilities in IDESIGN5[12]

Design optimization programs require information about the problem to be solved. This information includes: 1) input data, such as number of design variables, number of constraints, etc.; 2) cost and constraint functions; and 3) gradients of cost and constraint functions. If the gradient expressions are not supplied, the system produces them by the finite-difference method. When there is a mistake in the input data or problem definition, errors will occur in the solution procedure. The system gives explicit error messages to guide the designer to correct mistakes.

The system allows for interactive data entry. A menu is displayed for selection of proper data segment and entry. The system is protected from user's mistakes. If a data mismatch is found, messages are given in detail. The procedure is simple, so even a beginner can easily follow it.

It is extremely useful and important to monitor the optimal design process through the interactive session. Histories of the cost function, constraint functions, design variables, maximum constraint violation, and convergence parameter can be monitored. When the histories are graphically displayed, they are of great help in interactive decision making as well as for gaining insight into the design process. Design sensitivity coefficients of the cost function and potential constraints are displayed in the form of normalized bar charts. This information shows relative sensitivity of the design variables. If the design process is not proceeding satisfactorily (there could be inaccuracies or errors in the problem formualtion and modeling), it is necessary to stop it and check the formulation of the problem. This will save human as well as computer resources. Also, if one algorithm is not progressing satisfactorily, then a switch can be made to another one or the process can be restarted from any previous design. The system can give suggestions for design changes by using the design sensitivity coefficients. It is also possible to utilize the software in a batch environment. In this case, the system uses default values for the parameters found to be best through experience and numerical experimentations.

Specification of the Status of Design Variables

In the optimization process, some design variables have little effect on the final solution of the problem. Also, in many cases, some design variables reach their optimum value and have little effect on the iterative process after that point. Thus, we need to identify such design variables and fix them. In this way, the problem is defined by only a few important variables and its size is reduced. When a fixed variable becomes important in the design process, it can be released to the active group. The status of design variables can be defined interactively in IDESIGN5.

The capability to specify the status of the design variables can be exploited in practice to obtain a discrete optimum solution for the problem. When we design a structural or mechanical system, the member sizes are the design variables. Members must be selected from an available set, but design variables are considered continuous in the optimization process. This is one of the reasons that optimization has not been extensively used in practical engineering design. The difficulty can be overcome by defining the status of design variables. If a design variable becomes close to an available size, it can be fixed to that value interactively. This also means that the designer can use his intuition in selecting discrete member sizes. This is very important because an expert designer's experience is considered to be extremely valuable and reliable in the engineering design community. Until the optimum is found, design variables can be assigned fixed values nearest to the ones available. The graphical display of various data can help the designer in decision making.

An important point is *when* or *how* to determine the status of a design variable. The ideal solution would be to automatically determine the status using a sophisticated algorithm. This can be done by pattern recognition of the trend, just as an expert designer would do. However, such a solution requires knowledge of engineering capabilities that is a topic of future research. In IDESIGN5, status of a design variable can be changed only interactively.

Role of Data Base in IDESIGN5

In the design optimization process, large amounts of data must be generated and manipulated. Specifically, the interactive process needs enormous amount of data, so a proper data base and a data base management system (DBMS) are essential. In addition, since IDESIGN5 is composed of independent modules, a DBMS is needed to communicate between the modules. A data base for IDESIGN5 is designed and managed by a DBMS, as described in the next section. This systematic generation and handling of large amount of data, along with the modular structure, has greatly facilitated the coding and debugging of the system.

Data Base Management in IDESIGN5

The data base is a centralized collection of data accessible to several application programs and optimized according to a data base definition schema. The data base management system (DBMS) is a software program handling all access requests and transactions against the data base.

In practical design optimization, finite-element and other numerical methods are adopted during analysis of structural and mechanical systems. In general, the amount of data used in finite-element analysis is quite large and various schemes are used to reduce the storage of data. In addition, the formulation of constraints and the design sensitivity analysis are carried out using most of the data generated during the analysis phase. Interactive computations and graphics also require additional data for the display of the system model and intermediate results. Therefore, the data must be organized and saved properly in a data base for efficient design optimization.

Implementation of Data Base in IDESIGN5

In IDESIGN5, a data base management system called MIDAS/N is used for handling the data.[13] MIDAS/N stands for the management of information for design and analysis of systems/numerical model. MIDAS/N is an application-independent data base management system and uses a different approach to deal with data management problems of scientific and engineering computing. It is designed specifically to handle numerical data. A data base of MIDAS/N is stored in a file. It can be a direct or a sequential access file. The status of a data base can be temporary or permanent. Each data base contains several data sets, which can be either one- or two-dimensional arrays (vectors or matrices). The matrix can be rectangular or triangular. The data type of a data set can be character, short integer, long integer, real, or double precision real. Each data set has several access models, such as row, column, or submatrix. Any part of a data set can be accessed by just one call statement. The dimension of the data set can be redefined dynamically.

At the beginning of each module in IDESIGN5, the necessary data are retrieved from the data base. Generated data are stored in the data base during the execution of the module. Connection with the data base needs additional CPU time; IDESIGN5 is designed to minimize the additional time. Usually, design optimization data are in the form of matrices or vectors. One matrix is defined as a data set, and several vectors having the same dimensions are included in a data set. For interactive graphics, the history of the design data is stored in the data base and retrieved when it is used.

Design of a Data Base

Collection of Information

To design a data base for IDESIGN5, the flow of the optimization data for various algorithms must be studied. Various vectors, matrices, and control parameters that are either input or generated during the solution process must be identified. All of the data must be collected and properly grouped to define the data base. In every module, these data are retrieved from the data base and stored in a common array for further processing. Table 1 lists the data for all the modules. Data are stored or retrieved using the variable names shown in the table. The variable names start with different letters according to their dimensions. One or several optimization data are stored in a data set.

Data Set Definition

By the characteristics of the optimization data, the data sets can be defined. One data set represents a matrix or several vectors having the same dimensions. If several vectors are included in a data set, they are stored like a matrix. The list of data sets is shown in Table 2. In some cases, a data set is stored by row but retrieved by column, or vice versa. These are also shown in Table 2. A permanent data base FPERM is defined at the beginning of the execution of the system and used in each module.

Design of IDESIGN5

Structure of IDESIGN5

A general-purpose optimization software program can have an overall structure as shown in Fig. 1. Each block can be composed of several modules. Figure 2 shows the structure of IDESIGN5 and the sequence in which the modules are executed. Each block is a program that can be compiled and executed independently. As shown in Fig. 2, the modules have two libraries, IDESIGN5.LIB and SMART.[14] IDESIGN5.LIB contains some subroutines common to many IDESIGN5 modules. SMART is a library containing several general-purpose interactive aids and subroutines for matrix and other standard operations. Some modules are used in one algorithm, while others are shared by many. When the modules are combined into one system, a main program controls the flow of progress based on the decisions made in every module. When each module is used independently, the operating system capabilities are used for controlling the flow of the calculations.

Each module has a buffer array for vector and matrix data and it is distributed to each subroutine according to the needs. A common array is declared for the problem parameters used in all modules. At the beginning of a module, the problem parameters are retrieved from the data base.

Definition of Optimization Problem

In this block of Fig. 1, the designer has to provide two types of information to use the system: input data for the problem

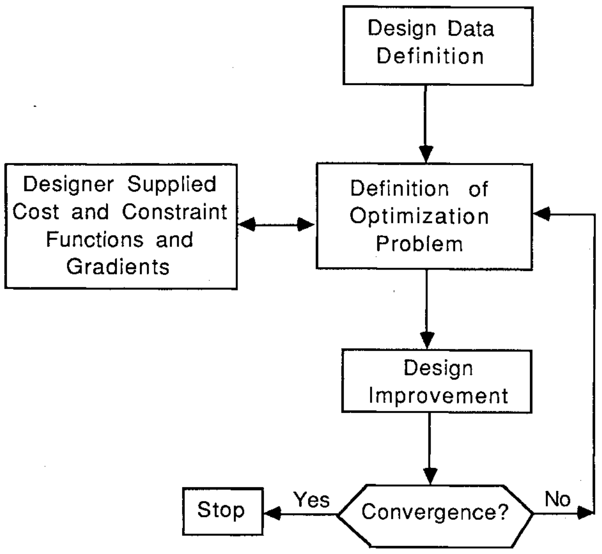
Design Data Definition
Designer Supplied Cost and Constraint Functions and Gradients
Definition of Optimization Problem
Design Improvement
Stop
Yes
Convergence?
No

Fig. 1 Overall structure of a design optimization program.

Table 1 List of data

Data description	Type	Dimension[a]	Variable name
Cost function gradient	Vector	nv	ACOGR
Values of design variables	Vector	nv	ADEVL
(Hessian matrix)* design change	Vector	nv	AHXDB
Increment of design variable	Vector	nv	AINCT
Gradient of Lagrangian at previous iteration	Vector	nv	ALOLD
Lower bound of design variable	Vector	nv	ALOWR
Upper bound of design variable	Vector	nv	AUPPR
Lagrange multipliers of QP subproblem	Vector	nv + nc	CLMQP
Constraint function value	Vector	nc	CONFN
Lagrange multipliers	Vector	nc	CORLM
Constraint gradient lengths	Vector	nc	CSENM
History of constraint functions	Matrix	itrs × nc	DCOFN
History of convergence parameter	Vector	itrs	DCONP
History of cost function	Vector	itrs	DCOST
History of design variables	Matrix	itrs × nv	DEIVL
History of maximum constraint violation	Vector	itrs	DMAXV
Active constraint set	Vector	nc	LC
Design variable status	Vector	nv	NFIX
Active set of previous iteration	Vector	nc	NPREC
Constraint sensitivity matrix	Matrix	nv × nc	TSENM
Hessian matrix	Vector	nv × (nv + 1) ÷ 2 + nv	UPPAT

[a]nv = number of design variables, nc = number of constraints, itrs = number of iterations.

Table 2 Structure of data base FPERM

Data set name	Included data (variable name)	Dimension[a]	Order of creation	Order of use
SDINV	ACOGR, ADEVL AHXDB, AINCT ALOLD, ALOWR AUPPR, NFIX NORDR	nv × 9	Col	Col
SDINC	CONFN, CORLM CSENM	nc × 3	Col	Col
SDINQ	CLMQP	nv + nc	Col	Col
SSENM	TSENM	nv × nc	Col	Col, row
SUPDT	UPPAT	nv(nv + 1) ÷ 2 + nv	Col	Col
SHISA	DCOND, DCOST DMAXV	itrs × 3	Col	Col
SHISB	DEIVL	itrs × nv	Col	Row
SHISC	DCOFN	itrs × nc	Col	Row
SINTG	LC, NPREC	nc × 2	Col	Col
SSYST	System parameters	300	Col	Col

[a]nv = number of design variables, nc = number of constraints, itrs = number of iterations.

and definition of the design problem. The input data for a problem can be provided interactively from a terminal through menu displays or from an input file. The definition of the design problem in Fig. 1 is provided by the designer through four independent modules. These are COSTFN, CONSTFN, COSTGR, and CONSTGR, as shown in Fig. 2. They contain the cost function, constraint functions, cost function gradient, and constraint function gradients, respectively. These modules can be very simple or very complex depending on the application. The modules calculate function values and gradients and store them in the data base. They can be written using any utilities of the computer. If the analytic expressions for the gradients are not available, IDESIGN5 can automatically calculate the gradients by the finite-difference method. In this case, the modules for gradient calculations need not be supplied.

Modules Suitable for Many Algorithms

The modules shared by every algorithm are as follows.

Input

This module is the first one executed in IDESIGN5 system. All input data for the design problem are defined. Input data consist of five groups: 1) initial terminal session—file definitions and terminal type; 2) problem definition—numbers of design variables and constraints, etc.; 3) parameter definition—convergence parameters, printing code, etc.; 4) design variable data—initial design, lower and upper bounds and status of design variables; and 5) algorithm selection. Each group of data is defined using a menu display. Between two consecutive menus, the designer can change the data already defined or the program can be terminated. Also, HELP facility is available at any point of process. Various menus used during the interactive session are given in Park and Arora.[12]

Interactive

By designer's option, the interactive module can be executed during any iteration. Features of this module are similar to the INPUT module. The designer can control the progress of the problem-solving procedure and problem data can be changed at any point in the execution. Proper menus are displayed for

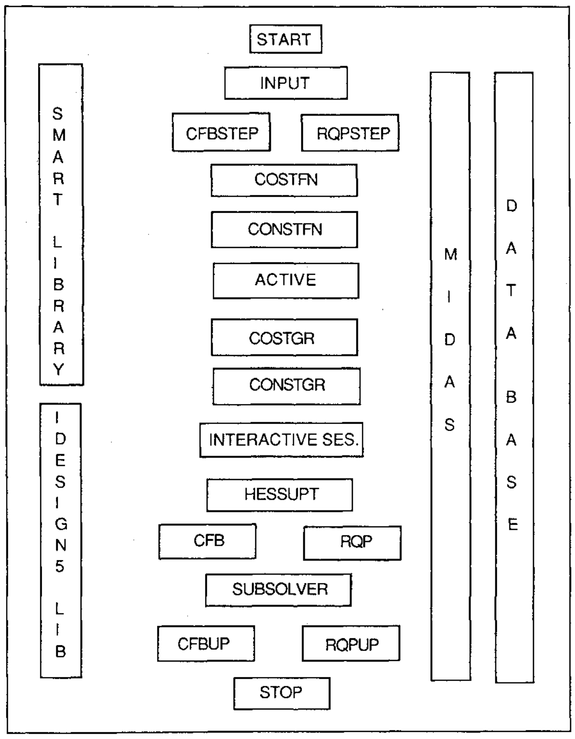

Fig. 2 Structure of IDESIGN5 and its modules.

changing the algorithm or the current data. To help the designer's decision, graphical displays for various data are available. The histories of parameters are plotted. Relative sensitivities for cost and constraint function gradients are shown as bar charts. Advice for design-variable change can be given using linear extrapolation. When the design point is in the infeasible region, the relation between the correction of constraint violation and cost function improvement is used to advise the designer by linear extrapolation of the sensitivity information.

Active

The potential constraint subset is defined in this module. The potential constraints identified here are used to define a subproblem that determines the search direction in the design space.

Subsolver

Normalized[4] subproblems are solved for the search direction using the subroutine QPSOL[15] in this module. Fixed design variables are dropped.

Modules for Problem Definition

As mentioned earlier, the COSTFN, COSTGR, CONSTFN, and CONSTGR modules should be prepared for cost function, cost function gradient, constraint functions, and constraint function gradients, respectively. Any data in the data base can be used, and the produced data can be stored in the data base.

Modules for LINRM and PLBA Algorithms

These modules are used in LINRM and PLBA methods and the hybrid method after a switch to the PLBA algorithm is made.

RQPSTEP

In this module, step size is determined by a one-dimensional search that requires evaluation of the cost and constraint functions. These are calculated using the designer-supplied modules.

RQP

Data that the subproblem will use to determine a search direction are generated.

HESSUPT

The Hessian matrix of the Lagrange function is updated in the PLBA algorithm in the factorized form. Through internal options, the Hessian can be set to an identity matrix and a direct update can be obtained when the full matrix is updated. When the condition number of the Hessian matrix is over a certain positive large number, it is set to the identity matrix.

RQPUP

From the output of the subsolver, the design increment vector is determined and covergence criteria are checked.

Modules for CFB Algorithm

These modules are used in the CFB algorithm and hybrid method before switching.

CFBSTEP

Here, the maxium constraint violation is calculated. From the second iteration onward, the step size determined in the CFBUP module is adjusted to remain within the bounds on the design variables.

CFB

Here, the data for the subproblem are calculated according to the current design conditions.

Table 3 Interactive execution of 25 bar truss design problem[a]

Iteration	Interactive change			
1	MVC = 2.7207			
	Algorithm = R3			
2	MVC = 0.4780			
	PLBA algorithm is used from this iteration			
	No.	Value	Given value	Status change
7	1	0.01	1.62	Fixed
	2	1.4831	1.80	Fixed
	4	0.0076	1.62	Fixed
	7	2.20765	3.13	Fixed
10	3	2.0093	2.88	Fixed
	7	3.13		Released
12	6	1.9958	2.38	Fixed
13	5	1.085928	1.80	Fixed
15	Optimum is found			
	MVC = 0.5693E − 06			
	ND = 0.3034E − 07			
	Cost function = 0.7744E + 03			
	CPU for data base = 29.015			
	Net CPU = 49.796			
	Total CPU = 78.811			
	No. of calls for function evaluation = 25			
	No. of total gradient evaluations = 163			
	Design variable values: 1.62, 1.8, 2.88, 1.62, 1.8, 2.38, 3.6295			

[a]MVC = maximum constraint violation, ND = norm of direction vector.

CFBUP

The design increment vector is calculated and the convergence criteria are checked.

Sample Applications

Performance of nonlinear programming algorithm can be ascertained only by numerical experiments requiring collection and implementation of the test problems. In IDESIGN5, the procedures of algorithms and tuning parameters are implemented through extensive numerical experiments.[12]

Many numerical example problems have been solved using the algorithms available in IDESIGN5. These include 115 test problems from the mathematical programming literature, small-scale engineering design problems, structural design problems such as trusses and frames, dynamic response optimization, and nonlinear response optimization. Details of all these applications are given in Ref. 12. Here we describe some details of applications to trusses and the interactive use of IDESIGN5.

The seven trusses, given in Refs. 16 and 17, are optimized for various constraint conditions, such as stress, displacement, and natural frequency. Combination of trusses and constraint conditions yields 15 problems. The formulation, design data, and sensitivity analysis have been discussed in the literatures.[3,6,17] The number of design variables are 7–47, state variables 8–150, and constraints 18–629. Multiple loading cases are treated.

For analysis of the structure and gradient calculation, a computer program TRUSSOPT is employed. TRUSSOPT is composed of four subroutines, each of which is attached to a designer-supplied module in IDESIGN5. TRUSSOPT does not use a data base management system.

Results for each problem in a summary form are given in Ref. 12. The data collected for each problem include the number of iterations, function evaluations, and gradient evaluations, CPU times for data base management and total execution, and optimum point. For the purpose of comparison, results of the PLBA algorithm in IDESIGN3 are also included. IDESIGN3 is the previous version of IDESIGN5 that does not use a data base management system. CFB fails on three problems and LINRM exceeds the maximum iteration limit in six problems. As expected, the performance of the hybrid and PLBA methods is superior to others. This was also observed in small-scale problem applications. The results of PLBA and the hybrid methods are similar to those of IDESIGN3. If we compare IDESIGN5 to IDESIGN3, the number of iterations increases in five problems and decreases in eight problems. The number of calls for function evaluations increases in six problems and decreases in eight problems. The total CPU time of IDESIGN5 is generally larger than that for IDESIGN3. Two main reasons cause the increase in IDESIGN5: 1) we need computing time for access to the data base and 2) the data are redefined at the beginning of each module. For example, the performances of PLBA and IDESIGN3 are exactly the same in the 25 bar truss with stress constraints except the computer time. For the 47 bar truss with all constraints, the number of calls for function evaluations with PLBA is greater than that of IDESIGN3. However, the total CPU time of IDESIGN5 is less than that for IDESIGN3. This is because the QP solver (QPSOL) in IDESIGN5 is more efficient than the QP solver (VEO4AD in the Harwell library) in IDESIGN3. Therefore, the algorithm and the software for solving the subproblem are also very important. When the structure becomes larger, the CPU time for the data base comprises a smaller percentage of the total CPU time. This is because the analysis needs a greater portion of the computing time and TRUSSOPT does not use a data base and DBMS.

Interactive Use of IDESIGN5

The 25 bar truss given in Refs. 16 and 17 is interactively optimized with all constraints. In practical optimization, the design variables are usually member sizes that must be selected from the available sections. Thus, discrete variable optimization should be employed. The present structure is designed to have double-angle sections given in the AISC code. The area of each member is considered as a design variable. The areas available in the AISC code are 1.62, 1.80, 1.99, 2.13, 2.38, 2.62, 2.63, 2.88, 2.93, 3.09, 3.13, 3.38, 3.47, 3.55, 3.63, 3.84, 3.87, 3.88, 4.18, 4.22, 4.49, and 4.59. The design process summarized in Table 3 is explained as follows:

Iteration 1

Maximum violation of constraints is very large, so constraint correction step R3 of CFB algorithm is chosen to correct constraints.

Iteration 2

Maximum violation is corrected; PLBA algorithm is selected from this iteration to the end.

Iteration 7

The design variables that are not changing during the iterative process are assigned values closest to those available. The history of the design variables is shown on a graphics terminal for this decision.

Iteration 10

Design variable 3 is fixed and design variable 7 is released. The sensitivity bar charts are exploited for this decision; that is, a sensitive design variable is released to the active status.

Iteration 12

Design variable 6 is fixed.

Iteration 13

Design variable 5 is fixed.

Iteration 15

The optimum is found. All the design variables have fixed values except the seventh. After optimum is found, design variable 7 is given the value 3.84. With these fixed values, there are no violated constraints. Therefore, these values can be taken as the final solution.

At the final point, all the design variables have discrete values. The cost function (0.774E+03) is higher than that (0.5907E+03) obtained earlier.[12] The reason is the lower bound used there is 0.01, while the smallest available size is 1.62. The optimum cost function with 1.62 as the lower bound and continuous design variables is 0.7161E+03 with the design as 1.62, 2.1478, 2.3341, 1.62, 1.9737, and 3.4309. Thus, the cost function increases by 8.1% when discrete design variables are used.

Use of IDESIGN5 with ADINA

The IDESIGN5 system has been also used to optimize truss structures with nonlinear response. A finite-element-method software system called ADINA is used for nonlinear analysis.[18] Different modules are developed for cost and constraint functions and sensitivity information. The formulation, analysis method, and design data are given in detail by Haririan et al.[19] Both geometric and material nonlinearities are included and many problems are solved. Thus, coupling of IDESIGN5 to a commercial analysis software has been successfully accomplished.

Discussion and Conclusions

A software system can be evaluated only by numerical tests. Numerous problems have been solved in batch as well as interactive environment, indicating that IDESIGN5 is a reliable software system. The conclusions based on the study are as follows:

1) A software system IDESIGN5 having diverse optimization capabilities has been developed. A data base for the system is designed and a data base management system is used to handle the data base. Procedures to integrate the data base management capabilities are developed and demonstrated. Use of a DBMS facilitated development of the system considerably.

2) The system can be readily expanded or combined with other systems. The combination of IDESIGN5 and ADINA illustrates this capability.

3) The algorithms in IDESIGN5 are implemented by numerical experimentation. Various problems have been effectively solved. In terms of the performance, PLBA and hybrid methods show better results.

4) Interactive execution shows good performance for the optimization process. Interactive change of problem parameters and design variable data can be exploited in problem solving. Experience with IDESIGN5 shows that a design optimization software must support interactive and graphics facilities for practical applications.

5) Discrete optimization is obtained by interactive definition of the status of the design variables. Expert designer's intuition can be thus exploited, making the system extremely useful in practical design optimization.

6) Data base and data base management systems have important roles in a powerful and user-friendly interactive design optimization system. The data base facilities can be used to collect raw data from which knowledge can be derived. Such knowledge can then be incorporated into an expert system that can act as a consultant and trouble-shooter during the design optimization process. This will be a useful extension of the system.[20]

7) Data base generation and management needs additional computational effort. However, as the problem size increases, the CPU time for data management decreases relative to the total CPU time.

Acknowledgment

This research is sponsored by the U.S. Air Force Office of Scientific Research, Air Force Systems Command under Grant AFOSR82-0322. The U.S. Government is authorized to reproduce and distribute reprints for governmental purposes, notwithstanding any copyright notation thereon.

References

[1]Arora, J. S. and Thandar, P. B., "Computational Methods for Optimum Design of Large Complex Systems," *Computational Mechanics*, Vol. 1, No. 2, 1986, pp. 221-242.

[2]Belegundu, A. D. and Arora, J. S., "A Recursive Quadratic Programming Method with Active Set Strategy for Optimal Design," *International Journal for Numerical Methods in Engineering*, Vol. 20, No. 5, 1984, pp. 803-816.

[3]Lim, O.K. and Arora, J. S., "An Active Set RQP Algorithm for Engineering Design Optimization," *Computer Methods in Applied Mechanics and Engineering*, Vol. 57, 1986, pp. 51-65.

[4]Arora, J. S., "An Algorithm for Optimum Structural Design Without Line Search," *New Directions in Optimum Structural Design*, edited by E. Atrek et al., Wiley, New York, 1984, Chap. 20.

[5]Vanderplaats, G. N., *Numerical Optimization Techniques for Engineering Design: With Applications*, McGraw-Hill, New York, 1984.

[6]Thanedar, P. B., Arora, J. S., and Tseng, C. H., "A Hybrid Optimization Method and Its Role in Computer-Aided Design," *Computer and Structures*, Vol. 23, No. 3, 1986, pp. 305-314.

[7]Gabriele, G. A. and Ragsdell, K. M., "Large Scale Nonlinear Programming Using the Generalized Reduced Gradient Method," *Journal of Mechanical Design*, Vol. 102, No. 3, 1980, pp. 566-573.

[8]Vanderplaats, G. N., Sugimoto, H., and Sprague, C. M., "ADS-1: A New General Purpose Optimization Program," AIAA Paper 83-0831, 1983.

[9]Balling, R. J., Parkinson, A. R., and Free, J. C., "OPT-DES.BYU: An Interactive Optimization Package with 2D/3D Graphics," *Recent Experiences in Multidisciplinary Analysis and Optimization*, edited by J. Sobieski, NASA CP 2327, 1984.

[10]Arora, J. S., Thanedar, P. B., and Tseng, C. H., "User's Manual for IDESIGN," Version 3.5, Optimal Design Laboratory, College of Engineering, University of Iowa, Iowa City, Tech. Rept. ODL-86.6, 1986.

[11]Atrek, E., Callagher, R. H., Ragsdell, K. M., and Zienkiewicz, O. C. (eds.) *New Directions in Optimum Structural Design: Proceedings of International Symposium on Optimum Structural Design*, 1981, Wiley, New York, 1984.

[12]Park, G. J. and Arora, J. S., "Interactive Design Optimization with Modern Database Management Concepts," Optimal Design Laboratory, College of Engineering, University of Iowa, Iowa City, Rept. ODL-86.9, 1986.

[13]SreekantaMurthy, T., Shyy, Y. K., and Arora, J. S., "MIDAS: Management of Information for Design and Analysis of Systems," *Advances in Engineering Software*, Vol. 8, No. 3, 1986, pp. 149-158.

[14]Arora, J. S., Lee, H. H., and Jao, S. Y., "SMART: Scientific Database Management and Engineering Analysis Routines and Tools," *Advances in Engineering Software*, Vol. 8, No. 4, Oct. 1986, pp. 194-199.

[15]Gill, P. E., Murray, W., Saunders, M. S., and Wright, M. H., *User's Guide for QPSOL*, Systems Optimization Laboratory, Dept. of Operations Research, Stanford University, Stanford, CA, Rept. SOL 84.6, 1984.

[16]Thanedar, P. B., Arora, J. S., Tseng, C. H., Lim, O. K., and Park, G. J., "Performance of Some SQP Algorithms on Structural Design Problems," *International Journal for Numerical Methods in Engineering*, Vol. 23, No. 12, 1986, pp. 2187-2203.

[17]Haug, E. J. and Arora, J. S., *Applied Optimal Design: Mechanical and Structural Systems*, Wiley-Interscience, New York, 1979.

[18]Bathe, K. J., "On the Current State of Finite Element Methods and Our ADINA Endeavours," *Advances in Engineering Software*, Vol. 2, No. 2, 1980, p. 59.

[19]Haririan, M., Cardoso, J. B., and Arora, J. S., "Use of ADINA for Design Optimization of Nonlinear Structures," *Computers and Structures*, Vol. 26, No. 1/2, 1987, pp. 123-134.

[20]Arora, J. S. and Baenziger, G., "Uses of Artificial Intelligence in Design Optimization," *Computer Methods in Applied Mechanics and Engineering*, Vol. 54, 1986, pp. 303-323.