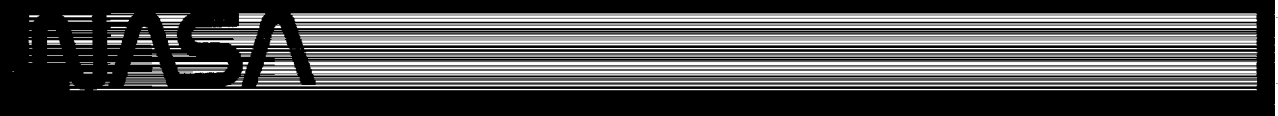
NASA Contractor Report 4413

Formal and Heuristic System Decomposition Methods in Multidisciplinary Synthesis

Christina L. Bloebaum

GRANT NAG1-1004
DECEMBER 1991



NASA Contractor Report 4413

Formal and Heuristic System Decomposition Methods in Multidisciplinary Synthesis

Christina L. Bloebaum
University of Florida
Gainesville, Florida

Prepared for
Langley Research Center
under Grant NAG1-1004

National Aeronautics and
Space Administration

Office of Management

Scientific and Technical
Information Program

1991

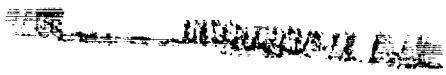


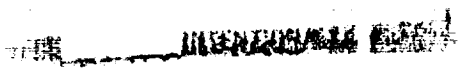
TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

ACKNOWLEDGEMENTS

I give my sincere thanks to Dr. Prabhat Hajela for the friendship and guidance he has given me during my graduate career. I owe much appreciation to Dr. Jaroslaw Sobieski, Head of the Interdisciplinary Research Office at NASA Langley Research Center, for his time and valuable suggestions, as well as for monitoring grants NAG 1-850 and NAG 1-1004. I am greatly appreciative of the time and assistance given me by Jim Rogers and Mike Riley of NASA Langley for their help with artificial intelligence and computer problems. I would like to acknowledge Bill LaMarsh and Laura Hall for the friendship they have given me, Joanne Walsh for her friendship and tolerance in allowing me the use of her computer, and the many people at NASA Langley who helped to make my stay there a productive and worthwhile one. Most importantly, I acknowledge the ever-present moral support and faith of my family.



CHAPTER 1
INTRODUCTION

Large scale engineering design problems are often characterized by multidisciplinary interactions in which participating disciplines are intrinsically linked to one another. The interdependencies of discipline analysis modules in such applications contributes to difficulties in successfully implementing a holistic design synthesis strategy. Furthermore, such an integrated implementation is also subject to complexities introduced as a result of an increased number of design variables and constraints. The objective of this work is to overcome the many obstacles inherent in the multidisciplinary problem in order to take advantage of the synergistic nature of integrated design.

When one speaks of design optimization, it is essential to distinguish between the analysis and design processes. Analysis involves determining the response of a defined system to its environment whereas design involves the process of defining that system [Van84]. The huge strides made in the development of structural analysis methods over the last forty years, combined with the growth of high power computing capabilities, has resulted in the increased application of optimization techniques in the design of engineering systems [Sch81].

The design process is initiated with a statement of requirements from which the design criteria are derived. Other design criteria are determined based on the design concept. The design process itself becomes a learning process as it is determined what the physics of the actual system can deliver in relation to the desirable system characteristics [Per84]. The actual design process encompasses several stages in which optimization methods could be applied in order to achieve an improved design. These stages, as described in Lem84, consist of the mission definition stage, in which system requirements

are defined, followed by the conceptual design, preliminary design, and finally, detailed design stages. The application of design optimization is most effective when introduced into the early stages of the design process, where numerous decisions must be made [Miu84].

The conceptual design phase produces some baseline configuration obtained as a result of complex trade-off studies. The process formerly consisted of guessing an initial configuration based on intuition and experience, analyzing the configuration, and then performing time consuming and tedious parametric studies to examine a prescribed design space. The quality of the answers was thus dependent on the skill of the designer [Lem84]. The application of optimization at this stage is especially useful in that an increased number of trade-off studies, design variables, and sophisticated analyses can be incorporated into the design process, thus aiding in the evaluation of competing design concepts.

The object of the preliminary design phase is to refine the design estimates made during the conceptual phase and to add additional detail to the configuration description. The design baseline is analyzed in significantly greater detail, involving simultaneous executions of discipline analyses among numerous design groups. As explained in Lem84, the simultaneous nature of this stage frequently results in inconsistent designs among groups due to the lack of a definable hierarchy from which iterative loops could be meaningfully established. The incorporation of optimization in this stage has two-fold benefits. First, it is in the preliminary design stage that the designer has the largest number of important options and decisions to make, thus providing the environment in which optimization techniques can be applied with greatest impact on computational efficiency. Secondly, the recent advances in system decomposition methods [Sob90] permit meaningful design optimization in these non-hierarchic environments, thus eliminating the problems introduced as a result of the lack of an identifiable system hierarchy.

The final stage in the design process is that of detail design. This stage is largely mechanical in nature, involving substantially more complex analyses. Detail design is concerned with local aspects such as joints and openings. By the time this stage is reached, optimization has very little impact on the overall design requirements.

The present study addresses the applications of optimization in the preliminary design stage in which the most capability for positive change exists. As previously stated, a major concern in this stage involves achieving an accurate and efficient mathematical representation of large engineering systems in order to perform meaningful design synthesis. Two basic solution strategies have been proposed for these highly coupled design problems. The first involves an adhoc decomposition in which the participating analyses of the various subsystems are performed in some prescribed order. In such an approach, the resulting design is dependent upon the order in which the analyses are implemented. The more desirable strategy is one which embraces parallel processing, in which each subsystem is examined simultaneously and with due consideration of all subsystem interactions [Wei86].

Multilevel decomposition methods provide a systematic approach for decoupling large complex systems into smaller, more tractable subsystems. These methods account for the interactions between the subsystems on the basis of a linear sensitivity analysis. In a majority of such efforts, the decomposition is governed either by an obvious hierarchy in the system, or on the basis of discipline if there is indeed a multidisciplinary interaction.

The present study develops three general decomposition approaches for optimization of large engineering systems that are applicable in problems where a distinct system hierarchy is difficult to identify. The methods are particularly applicable for multidisciplinary design problems which are characterized by closely coupled interactions among discipline analyses. Recent technological and computer developments in the areas of cumulative constraint representations [Haj82], sensitivity analysis for non-hierarchic systems [Sch76 and Haf80], optimal sensitivity analysis [Sob82], and distributed

computing capabilities [Rog81], provide the necessary components to create a decomposition methodology that allows for truly integrated synthesis and has the advantage of subsystem modularity. Such an advantage allows for implementation of specialized methods for analysis, computational efficiency, and the ability to incorporate human intervention and decision making in the form of an expert systems capability.

It is important to stress that the results of this investigation are not methods applicable to only a specific situation, but rather, are methodologies which can be used for a large class of engineering design problems in which the system is non-hierarchic in nature. Specifically, two automated, or formal, methods are developed to accomplish this purpose. The methods are referred to as the Global Sensitivity Equation (GSE) Method [Sob88b] and the Concurrent Subspace Optimization (CSSO) Method, which is largely based on a blueprint for generic system decomposition in non-hierarchic environments [Sob88a]. The modularity of the subsystems which exists in the CSSO is taken advantage of to create a methodology which allows for heuristics to be applied in an embedded expert systems capability. This approach is referred to as the Concurrent Subspace Optimization - Embedded Expert System (CSSO-EES) Method.

In the investigation of the applicability of the GSE method for large scale engineering problems, a multidisciplinary test environment is used involving the disciplines of structures, aerodynamics and performance, and flight mechanics. The objective of the synthesis process is to find the minimum weight configuration of a general aviation aircraft subject to design considerations from all disciplines.

The feasibility of the CSSO method is demonstrated through implementation of a verification procedure in which a simplistic ten-bar truss model provides the test bed. The minimum weight configuration is sought, with constraints and design variables stemming from topology determination and member sizing subsystems.

The applicability of an expert systems capability is investigated for the CSSO-EES method using a control/structure interaction problem. The object of the synthesis problem

is to determine the minimum weight design of a ten-bar truss subject to static and dynamic loadings, with constraints placed on static stress, natural frequencies, and static and dynamic displacements.

In this chapter, the application of optimization in the various design phases is introduced and some of the problems associated with these applications is discussed. The objectives of this study are then stated, with a brief description of the multidisciplinary example problems used for verification purposes.

Chapter 2 contains a review of literature pertinent to the basic understanding that is required in order to appreciate the development of the synthesis methodologies on which this dissertation focuses. The most crucial developments in the field of optimization are presented, including a review of actual industrial applications of optimization methods in the design process.

Chapter 3 focuses on the synthesis methodology required to implement optimization in a highly coupled environment. The difference between hierarchic and non-hierarchic systems is established. Basic concepts and definitions are introduced with a generic optimization statement.

A discussion of various approaches to determine coupled system sensitivity is presented in Chapter 4. The use of the Global Sensitivity Equation method is compared to other techniques, including a finite difference approach.

A generic development of the Global Sensitivity Equation (GSE) method, the Concurrent Subspace Optimization (CSSO) method, and the Concurrent Subspace Optimization - Embedded Expert System (CSSO-EES) method is presented in Chapter 5. The requirements of each method, as well as potential applications, are also examined.

Specific applications of these methods in test problems are described in detail in Chapter 6. The mathematical models, analysis requirements, and computational tools are delineated for each method.

The implementation of solution techniques for the three methods is described in Chapter 7. The Global Sensitivity Equation method applications center on strategies to increase efficiency and solution accuracy for large problems. The Concurrent Subspace Optimization method and its heuristic counterpart, the Concurrent Subspace Optimization - Embedded Expert System method, are assessed with the aim of determining their feasibility.

Results obtained from the implementation of the solution techniques described in the previous chapter are discussed in Chapter 8. Conclusions drawn from these discussions, and recommendations for further research are presented in Chapter 9.

CHAPTER 2
LITERATURE SURVEY

Structural optimization applications prior to 1960 were predominantly based on a simultaneous failure mode approach, wherein the inequality constrained weight minimization problem was converted to obtaining the solution to a set of nonlinear simultaneous equations. Shanley [Sha52] and Gerard [Ger56] applied this approach to the minimum weight optimal design of aircraft structural components subjected to compressive loads. Other applications involved the plastic collapse design philosophy which allowed for planar frame structural optimization problems to be formulated as linear programming problems [Hey51, Fou54, Pra56, and Liv56]. Perhaps the first person to recognize that certain structural optimization problems could be treated as nonlinear mathematical programming problems was Klein [Kle55], who recognized the importance of considering inequality constraints in the problem formulation. There is no argument, however, that the precursor to today's applications of optimization was Schmit's pioneering work in 1960 [Sch60] in which he set forth the structural synthesis concept. In this work, Schmit introduced the concept of coupling structural analysis and nonlinear mathematical programming to create an automated optimum design capability that was applicable for a broad class of structural systems.

The 1960's saw efforts focus in two main areas involving component type problems [Sch65, Kic68, Str69] and the development of structural synthesis programs based on coupling finite element analysis and nonlinear mathematical programming concepts [Gel66 and Kar68]. One of the most significant efforts during this time was Morrow and Schmit's work in 1968 [Mor68], involving the minimum weight design of

By the early 1970's it had become apparent that the mathematical programming approach to structural optimization resulted in inordinately large computational times, thus making the approach impractical for industrial applications [Gel71]. This realization provided the motivation for improving mathematical programming efficiency and renewing interest in optimality criteria methods. The early 1970s also saw the beginning of interdisciplinary design research [Gil72 and Ful73].

The introduction of approximation concepts [Sch74] in 1974 led to mathematical programming based structural synthesis methods [Haf76 and Sch76] that were markedly more efficient than their predecessors. The state-of-the-art today continues to build upon these early developments, specifically with the intent of increasing computational efficiency and versatility of applications.

Recent interest in the problems associated with multidisciplinary optimization is evidenced by an increased number of conferences, journals, and publications devoted to the subject. Numerous papers have been published recently which deal specifically with multidisciplinary optimization applications in such diverse areas as naval structural design [Dhi84 and Hug84], spacecraft design [Fer84], rotorcraft design [Miu84], automobile design [Pra84], and aircraft design [Sen88]. The proposed methodologies to deal with the multidisciplinary design problem have been almost as diverse as the applications and have mostly proved disappointing.

The intuitive practice of breaking a large task into smaller, more manageable tasks was applied in Sobieszczanski-Sobieski [Sob82a] in which a linear decomposition method was applied for hierarchic environments only. Early attempts to solve the non-hierarchic problem involved wrapping an optimization loop around the contributing disciplinary analyses [Kro88]. Unfortunately, the approach was computationally prohibitive and tended to exclude human intervention and decision-making. The Global Sensitivity Equation method demonstrated in Sob90, Sob88b, and Blo87 extended the modularity

concept of Sob82 to include applications in the non-hierarchic environments existing in multidisciplinary problems and represented the state-of-the-art in the field as of 1990.

Two review papers in the field of multidisciplinary synthesis are particularly noteworthy. The requirements and opportunities available in multidisciplinary analysis and synthesis applications are reviewed in Tolson [Tol85]. The potentials and achievements of multidisciplinary optimization are reviewed in Sobieszczanski-Sobieski [Sob89].

CHAPTER 3
DESIGN PROBLEM STATEMENT AND METHODOLOGY

Design Problem Statement

Optimal design is concerned with achieving the best design according to some prescribed criteria while satisfying certain associated restrictions. Wilde [Wil78] defines the optimal design as being the best feasible design determined by some prescribed quantitative effectiveness measure. The motivation behind optimization applications is to exploit the available limited resources in such a way as to maximize output [Haf90]. As an example, a typical objective of an optimization application in the field of structural design is to determine the minimum weight structural configuration subject to restrictions on stresses and displacements. The importance of minimum weight design of structures is especially crucial to the aerospace industry where aircraft designs are controlled more by weight considerations than by cost.

The concept of optimizing a structure implicitly suggests that there is some freedom to change the structure. This is accomplished by changing a given set of design variables over some prescribed range. Design variables can be either discrete or continuous in nature. A continuous design variable has some range of variation in which it can assume any value; a discrete design variable can only assume a value from a specified list of potential values. A change in the design variables results in some change in the overall design response, either in the objective function or in the problem constraints.

The objective function is essentially a merit function that has some explicit or implicit relation to at least a subset of the design variables, and can be improved through manipulation of those variables. In a structural optimization problem, for example, the objective function would be structural weight, which would be influenced by design

variables associated with structural member sizes. Typically, in realistic optimization applications, limits exist on both design variables and some response functions that are dependent on at least a subset of the design variables. These limits are referred to as constraints. Upper and lower limits on the design variables are side constraints. Constraints which impose upper or lower limits on the quantities that are dependent upon a subset of the design variables, are by their very nature inequality constraints. Limitations which place exact value requirements on these quantities are referred to as equality constraints.

The notation that is adopted in this work for the objective function, constraints and design variables is demonstrated in the following optimization problem formulation.

$$\begin{array}{lll} \text{Minimize} & F(X) & \\ \text{Subject to} & g_j(X) \le 0 & j=1,\ldots l \\ & h_k(X) = 0 & k=1,\ldots m \\ \text{and} & X_i^L \le X_i \le X_i^U & i=1,\ldots n \end{array} \qquad (3.1)$$

where (X) represents a vector of design variables, g_j and h_k are inequality and equality constraints, respectively, and F is the objective function.

It is typical to normalize constraints in order to minimize potential mathematical problems associated with wide variations in orders of magnitude. This is accomplished through manipulation of the allowable limits that are placed on the response quantities of interest. For example, a constraint might exist such that the calculated lateral tip displacement of a cantilever beam must be less than a prescribed allowable limit. The inequality constraint would be formulated (according to the representation of Equation 3.1) as

$$u - u_{al} \le 0 \qquad (3.2)$$

where u is the displacement and u_{al} is the allowable limit. It is obvious that this constraint formulation will have units associated with distance. To place the constraints on the same basis so that constraint values are all of order one, a normalized representation can be made as follows.

$$\frac{u}{u_{al}} - 1 \leq 0 \tag{3.3}$$

The normalized constraint representation of Equation 3.3 will be used throughout this work.

In the complex engineering design problem associated with a multidisciplinary application, contributions to the design variables, constraints, and the objective function are made from all the participating disciplines. The design variable and constraint vectors can then be described in terms of partitioned vectors, where partitioned subsets are associated with each discipline's contributions.

Synthesis Methodology

The general solution process for a gradient-based optimization problem can be seen in Figure 3.1. The process begins with an initialization of design variables and problem parameters from which an analysis is performed. A sensitivity analysis is then carried out to find the first derivative information of the output response quantities, such as the objective function and constraints. This sensitivity information is then used in the gradient-based optimizer which results in an improved value of the objective function. The process is terminated when no further improvement in the objective function can be made without violating the constraints. In a non-hierarchic environment, however, the design process necessarily changes.

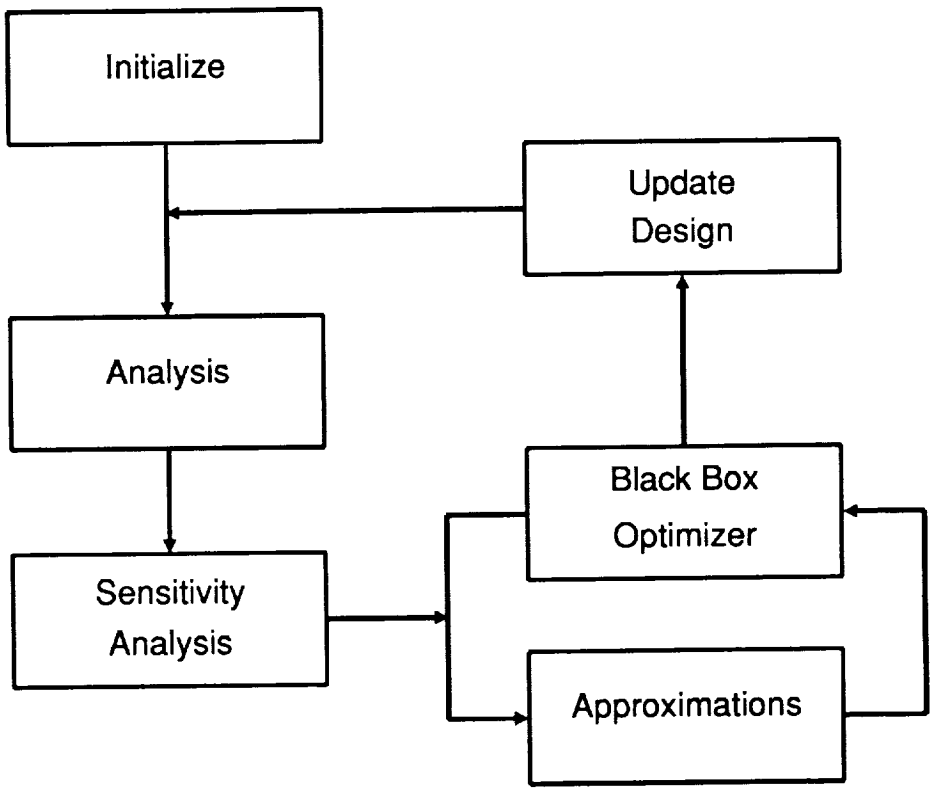


Figure 3.1 Generic design methodology for gradient-based optimization.

A non-hierarchic system is one in which the interactions between subsystem modules cannot be distributed in a top-down hierarchy such as that demonstrated in Figure 3.2a. Non-hierarchic systems are characterized by subsystem analyses that are linked through transference of output data, creating a complex network like that of Figure 3.2b. The synthesis methodology for such a non-hierarchic system is shown in the flowchart in Figure 3.3. The intrinsically linked subsystem analyses must first be performed within an iterative framework in order to obtain a converged initial point. A converged initial point is defined as that point which satisfies the equations

$$\begin{aligned} &SS1=0 \\ &SS2=0 \\ &\vdots \\ &SSN=0 \end{aligned} \tag{3.4}$$

where SSi corresponds to the analysis associated with the ith subsystem. Once a converged initial point is obtained, the sensitivity analysis can be performed. However, due to the large number of analyses required in the process, computational expenses are often exorbitant. The available computer tools used to perform analysis in such complex environments, such as structural or aerodynamic analyses, are inevitably computationally expensive. The piecewise linear optimization approach, or method of approximate programming, is extremely useful in reducing these computational expenses.

In the method of approximate programming [Gri61], gradient information is used to create an approximate optimization problem that is solved in lieu of the fully nonlinear problem, thus reducing repeated costly analyses [Sch76c]. The optimization is then carried out in the neighborhood of the current design point. Move limits are imposed on the user prescribed design variables during the optimization process; this is required in order to maintain the integrity of the linear approximations of the output response quantities. Determination of move limit values is generally based on problem-dependent heuristics and user experience.

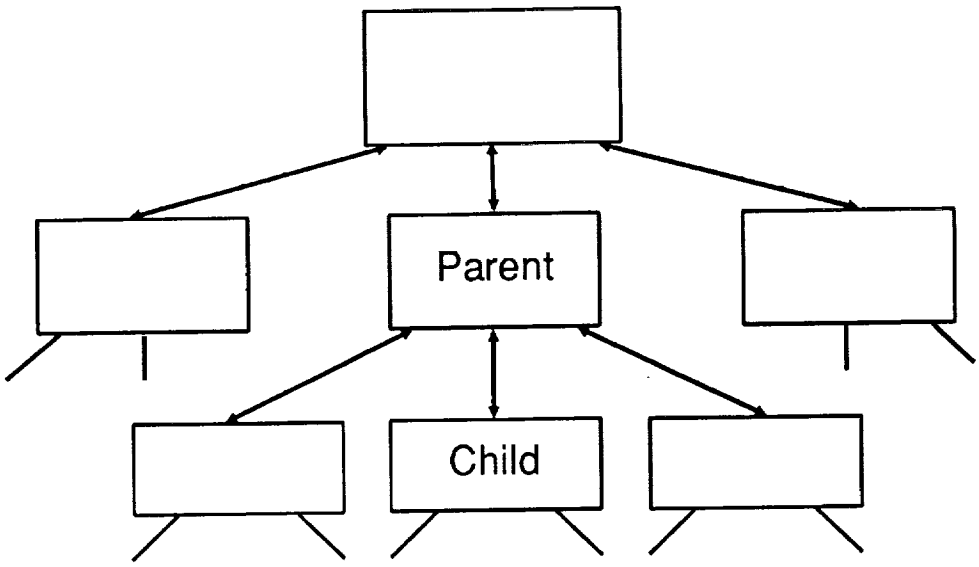


Figure 3.2a Hierarchic system decomposition network.

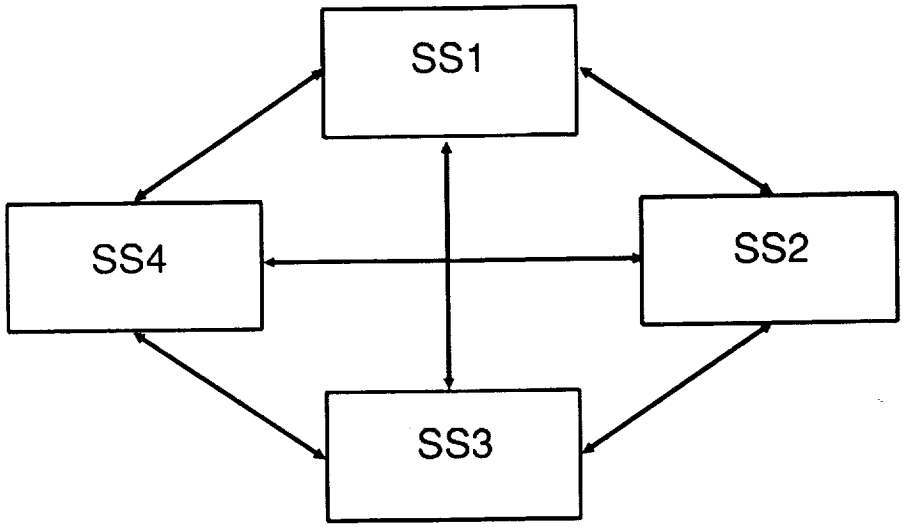


Figure 3.2b Non-hierarchic system network representing subsystem interactions.

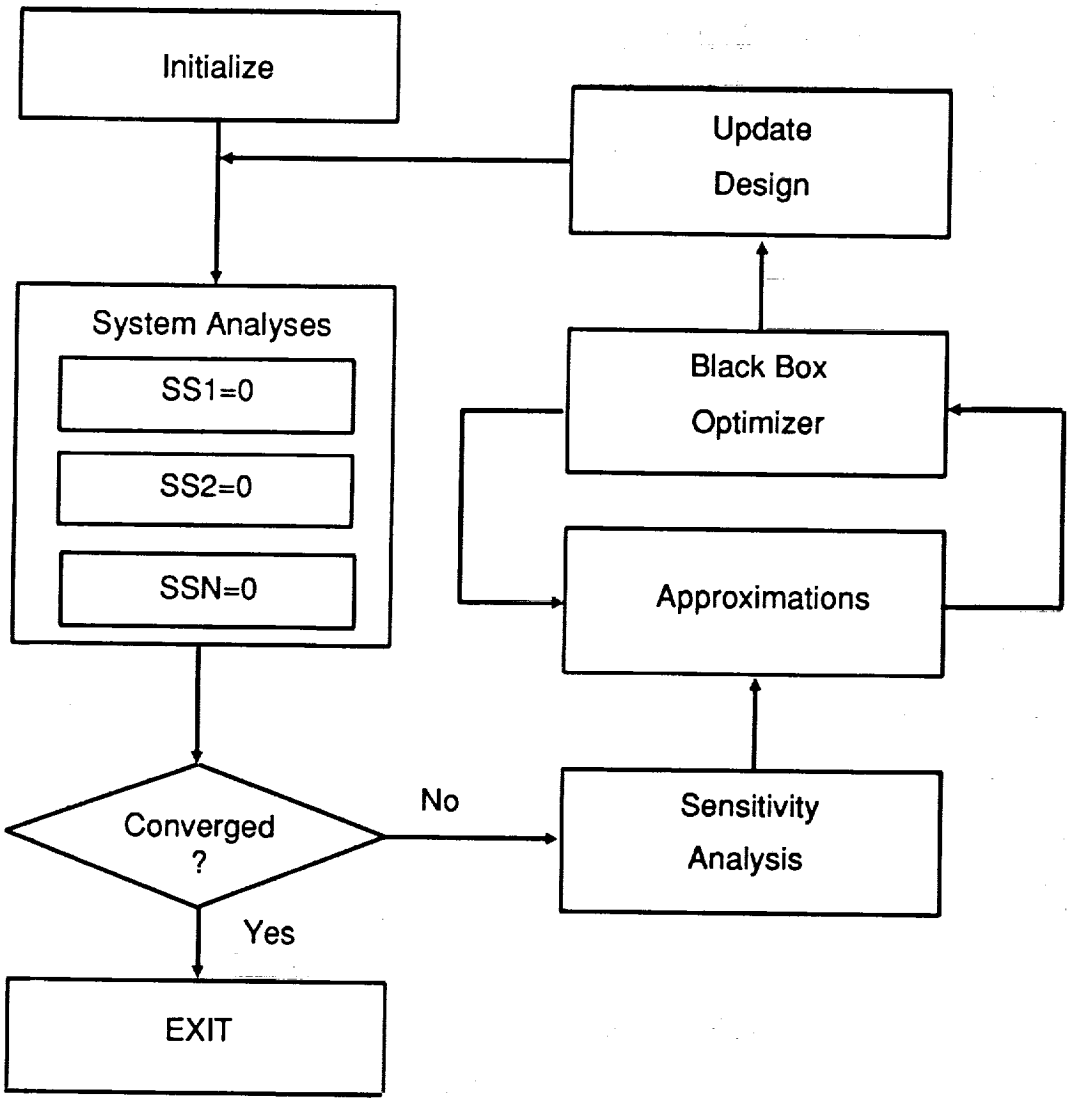


Figure 3.3 Design synthesis methodology for generic non-hierarchic multidisciplinary problems.

Although several approximation techniques have recently been investigated for application in this methodology [Sto74, Sta79, and Fad90], one of the most popular and easiest to implement is based on a first order Taylor series expansion of the objective function and constraints. Due to the fact that this information is already available for use in the gradient-based optimizer, no additional effort is required. The linearized optimization problem based on this type of approximation is of the form

$$\begin{aligned} \text{Minimize} \quad & \left\{F(\hat{\mathbf{X}}) + (\mathbf{X}-\hat{\mathbf{X}})^T \nabla F(\hat{\mathbf{X}})\right\} \\ \text{Subject to} \quad & \left\{g_j(\hat{\mathbf{X}}) + (\mathbf{X}-\hat{\mathbf{X}})^T \nabla g_j(\hat{\mathbf{X}})\right\} \le 0 \\ & \left\{h_k(\hat{\mathbf{X}}) + (\mathbf{X}-\hat{\mathbf{X}})^T \nabla h_k(\hat{\mathbf{X}})\right\} = 0 \\ \text{and} \quad & \hat{X}_i - \alpha_i \le X_i \le \hat{X}_i + \beta_i \end{aligned} \tag{3.5}$$

where α and β are prescribed positive constants called move limits and $\hat{\mathbf{X}}$ is the design point about which the objective function and constraints are linearized. These move limits effectively serve to limit the range of variation of the design variables. The optimal design resulting from the approximate optimization problem of Equation 3.5 then forms the initial point for the next cycle. The process is terminated when prescribed convergence criteria are met.

For a highly nonlinear problem, it is essential that appropriate move limits be established [Mor82]. By allowing the design variables to change only within some percentage of the initial point, the inaccuracies introduced due to the linear approximations are effectively controlled. An example of limiting the movement of the design variables in this manner can be seen in Figure 3.4. A two-dimensional design space is shown with lines of constant objective function and the constraint boundary defined. A larger move limit, as seen in case B, results in a greater error than that associated with case A due to the linear approximations.

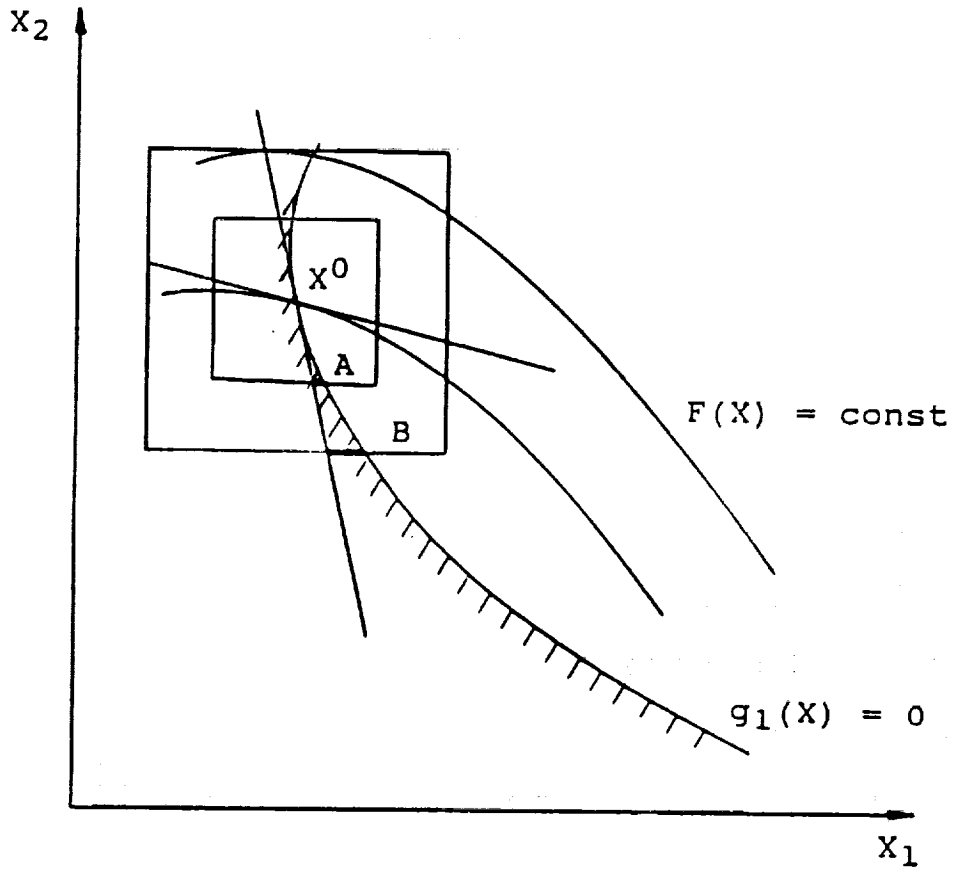


Figure 3.4 Effect of limiting design variable movement in two-dimensional design space.

The need for restrictive move limits in certain optimization applications can be demonstrated in the design of a rectangular beam for minimum weight subject to stress limitations, where the design variables are the depth and width of the rectangular cross-section. The displacement, load, and stress relations will be represented as scalars in order to demonstrate proportionality relationships. The load-displacement relation can be expressed,

$$K\,u = P \tag{3.6}$$

where P is the applied load, u is the displacement and K is the stiffness. The stiffness can be replaced by (c I) to obtain

$$(c\,I)\,u = P \tag{3.7}$$

where c is a constant and I is the moment of inertia. The displacement is now expressed,

$$u = P / (c\,I) \tag{3.8}$$

Stress is defined in terms of a constant, S, and the displacement as follows.

$$\sigma = S\,u \tag{3.9}$$

Substituting the previous expression for displacement in Equation 3.9 yields

$$\sigma = S\,P / (c\,I) \tag{3.10}$$

The moment of inertia for the rectangular section is defined in terms of the width (w) and depth (d) as,

$$I = (1/12)\,w\,d^3 \tag{3.11}$$

Substituting this expression into Equation 3.10 yields a relation for stress in terms of the design variables as,

$$\sigma = 12\ S\ P / (c\ w\ d^3) \qquad (3.12)$$

From this expression it can be seen that the stress is proportional to the inverse of one design variable and the inverse of the cube of the other. This results in nonlinearities requiring small move limits to preserve the validity of assuming a linear behavior in the stress response.

CHAPTER 4
SENSITIVITY DETERMINATION

Sensitivity Analysis Overview

The first step in the analysis of a complex structure involves a discretization of the continuum equations into a finite difference, a finite element, or similar model. The analysis problem then requires the solution of algebraic equations, ordinary differential equations, or eigenvalue problems, depending on the response quantities involved. Determination of sensitivity required in the optimization involves a mathematical problem of obtaining the derivatives of those equation's solutions with respect to their coefficients [Haf90].

The sensitivity analysis is typically the most computationally expensive aspect of the optimization process. It is therefore essential that efficient approaches for sensitivity evaluation be used in the design process. Numerous techniques exist for the evaluation of these derivatives, with one of the most popular being the finite difference approach. Unfortunately, this approach is the most computationally expensive and often has accuracy problems. Other techniques that are commonly used are the analytical and semi-analytical approaches. A recently developed approach is the Global Sensitivity Equation (GSE) method [Sob90], which has significant advantages in complex engineering problem applications. These approaches will be reviewed in the following sections.

Finite Difference Approach

The finite difference approach is one of the most popular techniques for determining sensitivity information due to the simplicity of implementation. However, the approach is

computationally expensive and is often plagued with accuracy problems. The simplest finite difference approximation is the first-order forward difference approximation. Given some function u(x) of a design variable x, the forward difference approximation ($\Delta u/\Delta x$) to the total derivative (du/dx) is

$$\frac{\Delta u}{\Delta x} = \frac{u(x+\Delta x) - u(x)}{\Delta x} \tag{4.1}$$

Another commonly used difference approximation is the second-order central difference approximation expressed as

$$\frac{\Delta u}{\Delta x} = \frac{u(x+\Delta x) - u(x-\Delta x)}{2\Delta x} \tag{4.2}$$

The finite difference approach requires perturbing the design variable by some prescribed amount, determining the function value associated with that perturbation, and then formulating the approximation according to Equation 4.1 or 4.2. Accuracy problems associated with this formulation are due to truncation and condition errors. When the finite difference approach is used to determine sensitivities for a complex coupled engineering problem, the synthesis methodology is modified to include an outer convergence loop as seen in Figure 4.1.

One of the major flaws of the finite difference approach is the possibility that the effect of a small perturbation may be lost when filtered through a set of analyses iteratively. For example, in a space truss with over five hundred structural members, is it really possible to determine the variation in the tip displacement due to a 1% change in the area of a member at the root? If large perturbations are used to avoid this problem, it is possible that nonlinearities would yield imprecise sensitivity information.

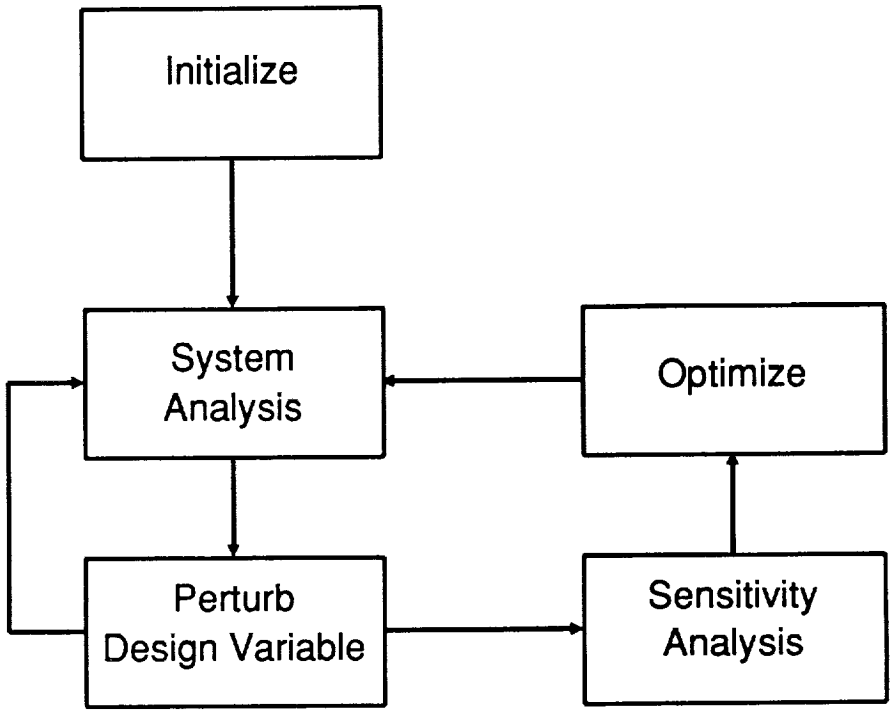


Figure 4.1 Design synthesis flowchart using Finite Difference approach.

Analytical Approach

Analytical derivatives are based on obtaining algebraic or differential equations for the sensitivities by analytically differentiating the governing equations. Although this approach is advocated by most researchers, the implementation is often difficult, especially when the functions under consideration have an implicit dependence on the design variables, and even that, in many cases, is enveloped in complex software packages. The modifications required to obtain analytical derivatives in such circumstances are often extremely difficult, and sometimes result in computational costs which exceed even the finite difference approach.

Implementation of the analytical method for the determination of first-order derivatives of static displacements is as follows. The equilibrium equations are generated from a finite element model in the form

$$[K]\ \{u\} = \{P\} \tag{4.2}$$

where [K] is the stiffness matrix, {u} is a vector of nodal displacements, and {P} is a load vector. A static displacement constraint can be expressed in terms of the nodal displacements and a design variable, x, as

$$g(\{u\},x) \leq 0 \tag{4.3}$$

Applying the chain rule of differentiation yields the expression

$$\frac{dg}{dx} = \frac{\partial g}{\partial x} + \frac{\partial g}{\partial \{u\}} \frac{d\{u\}}{dx} \tag{4.4}$$

The first term on the right hand side is generally zero, leaving only the second term which must be evaluated. Differentiation of Equation (4.2) with respect to x yields the expression

$$[K]\frac{d\{u\}}{dx}=\frac{\partial\{P\}}{\partial x}-\frac{d[K]}{dx}\{u\} \tag{4.5}$$

Premultiplication of both sides of Equation (4.5) by the term

$$\frac{\partial g}{\partial\{u\}}[K]^{-1}$$

yields the expression

$$\frac{\partial g}{\partial\{u\}}\frac{d\{u\}}{dx}=\frac{\partial g}{\partial\{u\}}[K]^{-1}\left(\frac{\partial\{P\}}{\partial x}-\frac{d[K]}{dx}\{u\}\right) \tag{4.6}$$

The solution of Equation (4.6) can be achieved by either the direct or adjoint method [Haf90] yielding the analytical derivative for the constraint $g(\{u\},x)$.

Semi-Analytical Approach

The semi-analytical approach, as the name implies, involves a combination of analytical and non-analytical methods. The analytical approach requires derivatives of the stiffness matrix and load vectors with respect to the design variables. These derivatives are often extremely difficult to obtain, especially when complex software packages such as finite element programs are used. In the semi-analytical approach, the derivatives of the stiffness matrix and load vectors are approximated by finite differences. The derivative of the stiffness matrix with respect to a design variable, x can be approximated by a forward finite difference representation, for example, as

$$\frac{d[K]}{dx}\approx\frac{[K(x+\Delta x)]-[K(x)]}{\Delta x} \tag{4.7}$$

A similar expression can be written for the derivative of the load vector with respect to the design variable. However, in typical static structural analysis problems, this derivative is generally zero.

Although the semi-analytical approach is as efficient as the analytical approach, due to the inclusion of approximations made by application of the finite difference technique, the resulting derivatives are subject to some of the same accuracy problems associated with the finite difference derivatives.

Global Sensitivity Equation Approach

The Global Sensitivity Equation (GSE) approach involves defining total derivatives of the output response quantities in terms of local sensitivities of the outputs of each subsystem with respect to that subsystem's inputs. Although the local sensitivities are determined by a finite difference approach, these sensitivities are calculated within each subsystem, thus removing the need of an outer iterative loop that would introduce unacceptable inaccuracies into the solution. The method is particularly applicable for complex engineering problems in which numerous coupled subsystems exist. A detailed development of the GSE Method is presented in Chapter 5.

CHAPTER 5
FORMAL AND HEURISTIC SYSTEM DECOMPOSITION METHODS

A generic development of the Global Sensitivity Equation (GSE) method, Concurrent Subspace Optimization (CSSO) method, and Concurrent Subspace Optimization - Embedded Expert System (CSSO-EES) method is presented.

Formal Methods

Two methods are introduced that provide a vehicle for automated design of complex, coupled engineering systems. Both methods are strictly algorithmic in nature, making use of problem-dependent heuristics only to the extent of initializing parameters prior to implementation of the design process. A development of these two formal approaches to system decomposition for design and optimization is presented.

Global Sensitivity Equation Method

The Global Sensitivity Equation approach is a methodology for obtaining the total sensitivity of the output response quantities of each subsystem with respect to the design variables of each subsystem. The total derivative information thus obtained is utilized in constructing the approximate optimization problem described in Chapter 3. The design variables and constraints from each subsystem are considered at the system level in an 'all-in-one' optimization within the context of the piecewise linear approach.

The underlying concepts in this formal approach for decomposition are simple and make use of the fact that the first derivative of a nonlinear function at a point is equal to the

first derivative of the function linear approximation at that point. Consider the case of two disciplines A and B, the interactions between which are illustrated schematically in Figure 5.1. The analysis equations for the two disciplines can be expressed in a symbolic form as follows.

$$\begin{aligned} A((X_A, Y_B), Y_A) &= 0 \\ B((X_B, Y_A), Y_B) &= 0 \end{aligned} \tag{5.1}$$

Here, X_A and X_B are the variables local to the system A and B, respectively. Y_A is the output vector for the system A and, in the most general form of coupling, this vector acts as a set of auxiliary input variables for system B. Similarly, Y_B is an auxiliary set of input variables for system A. Thus, the variables Y_A and Y_B provide the coupling between the two systems. It is possible to rewrite the above expressions in an explicit form as follows.

$$\begin{aligned} Y_A &= (X_A, Y_B) \\ Y_B &= (X_B, Y_A) \end{aligned} \tag{5.2}$$

A first order Taylor series representation allows us to write,

$$\begin{aligned} \frac{dY_A}{dX_A} &= \frac{\partial Y_A}{\partial X_A} + \frac{\partial Y_A}{\partial Y_B}\frac{dY_B}{dX_A} \\ \frac{dY_B}{dX_B} &= \frac{\partial Y_B}{\partial X_B} + \frac{\partial Y_B}{\partial Y_A}\frac{dY_A}{dX_B} \end{aligned} \tag{5.3}$$

The chain rule can be applied to Equation 5.2 to obtain two more equations of the following form.

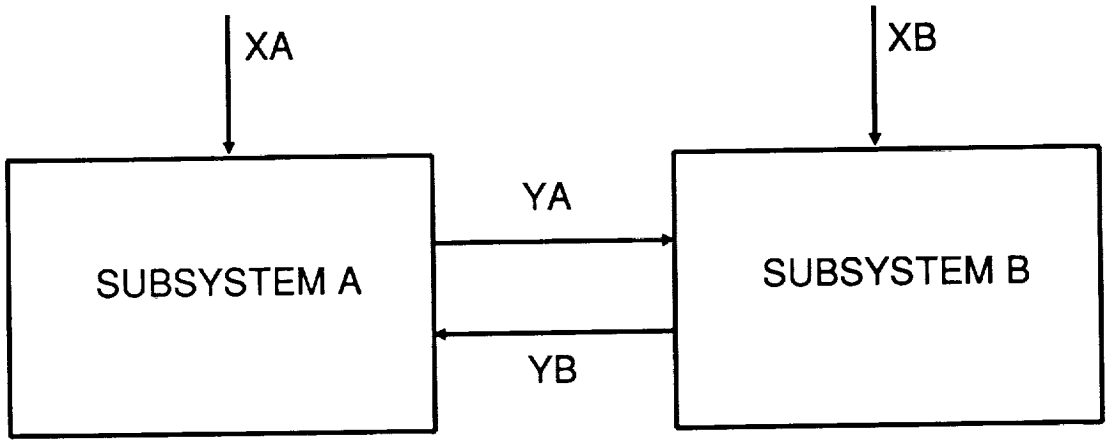


Figure 5.1 Subsystem interactions flowchart.

$$\frac{dY_A}{dX_B}=\frac{\partial Y_A}{\partial Y_B}\frac{dY_B}{dX_B}$$

$$\frac{dY_B}{dX_A}=\frac{\partial Y_B}{\partial Y_A}\frac{dY_A}{dX_A} \qquad (5.4)$$

These equations can be represented in a matrix notation as follows.

$$\begin{bmatrix} I & -\frac{\partial Y_A}{\partial Y_B} \\ -\frac{\partial Y_B}{\partial Y_A} & I \end{bmatrix} \begin{Bmatrix} \frac{dY_A}{dX_A} \\ \frac{dY_B}{dX_A} \end{Bmatrix} = \begin{Bmatrix} \frac{\partial Y_A}{\partial X_A} \\ 0 \end{Bmatrix}$$

$$\begin{bmatrix} I & -\frac{\partial Y_A}{\partial Y_B} \\ -\frac{\partial Y_B}{\partial Y_A} & I \end{bmatrix} \begin{Bmatrix} \frac{dY_A}{dX_B} \\ \frac{dY_B}{dX_B} \end{Bmatrix} = \begin{Bmatrix} 0 \\ \frac{\partial Y_B}{\partial X_B} \end{Bmatrix} \qquad (5.5)$$

Note that the total derivatives dY_A/dX_A, dY_A/dX_B, dY_B/dX_a, and dY_B/dX_B can be solved from the above set of equations if the partial sensitivity derivatives that appear in the coefficient matrix and in the right hand vector are known. These partial sensitivities can be computed locally within the system, eliminating the need to perform computationally expensive interdisciplinary iteration. This also diminishes the possibility of errors associated with round-off and truncation in the iterative process, from having adverse effects on the quality of the sensitivity results. It is worthwhile to note that the output from the analysis of one discipline may contain data that has no influence on other disciplines. As an example, the output from a structures analysis may include modal and frequency information that is passed as input to both aerodynamics and flight mechanics disciplines. However, it may also include data such as the objective and constraint function information that is not passed as input. Although this data has no influence on the analyses of the other disciplines, including it in the output vector yields total derivative information directly from

the solution of the GSE. Post-processing to determine the derivatives of these output response quantities is, thus, unnecessary.

Since the Global Sensitivity Matrix (GSM) is a function of local sensitivities of outputs to inputs which can be obtained within each discipline independently, the approach essentially permits a decoupling of large systems into smaller subsystems. The sensitivities obtained from the above analysis can be used to develop linear approximations to the output response of each subsystem, which can be subsequently employed in the gradient-based piecewise linear optimization process. However, due to the complexities of large engineering problems, the dimensionality of the local sensitivity matrices may be prohibitively large for repetitive decomposition in an optimization sequence, and may contribute to substantial reductions in the numerical efficiency.

Concurrent Subspace Optimization Method

The Concurrent Subspace Optimization (CSSO) method permits the decoupling of a large engineering system into smaller subsystem modules in order to achieve concurrent optimizations in each of these subspaces. This method essentially takes the concept behind the Global Sensitivity Equation method one step farther, performing not only the sensitivity analyses within each individual subsytem, but the optimizations as well. Unlike the conventional method of subspace optimizations, however, the proposed method eliminates the need for a full analysis in each subspace, thereby providing potential computational savings. The method is particularly well-suited to applications in a design organization setting in which tasks are distributed among groups of specialists representing physical subsystems and disciplines.

The evolution of optimization techniques has resulted in quite diverse and largely discipline-dependent approaches. Certain algorithms are often totally dependent upon the unique physics associated with the discipline in question. For instance, optimality criterion

methods are specifically tailored for structural weight minimization applications. Hence, certain of these discipline-dependent optimization techniques may not be applicable within the framework of a traditional system level optimization approach based on sensitivity information. The main motivating factor behind use of the CSSO, therefore, is the ability to take advantage of the discipline-dependent algorithms within each subspace optimization.

Implementation of the CSSO progresses as seen in Figure 5.2. A system analysis is first performed in which contributing analyses, or subsystems, are first defined in order to obtain behavioral response sensitivities by application of the GSE. Constraints in each subspace are represented by a single cumulative constraint measure, C, by means of a Kresselmeier-Steinhauser (K.S.) function [Haj82].

The cumulative constraint can be written

$$C = \frac{1}{\rho} \ln \left[\sum_{j=1}^{m} \exp(\rho \cdot g_j) \right] \tag{5.6}$$

where m is the number of constraints being represented in the cumulative constraint formulation and ρ is a user-prescribed constant. A smaller value of ρ allows more constraints to participate in the cumulative constraint representation while a larger value of ρ allows the most critical constraint to dominate. The derivative of this representation with respect to the design variable X_i may be determined analytically as follows.

$$\frac{dC}{dX_i} = \left[\sum_{j=1}^{m} \exp(\rho \cdot g_j) \right]^{-1} \left[\sum_{j=1}^{m} \left\{ \frac{dg_j}{dX_i} \exp(\rho \cdot g_j) \right\} \right] \tag{5.7}$$

The design variables are then allocated to the subspaces on which they have the greatest impact. This allocation is based on the sensitivity of the cumulative constraint and on the sensitivity of the objective function with respect to the design variables in the form

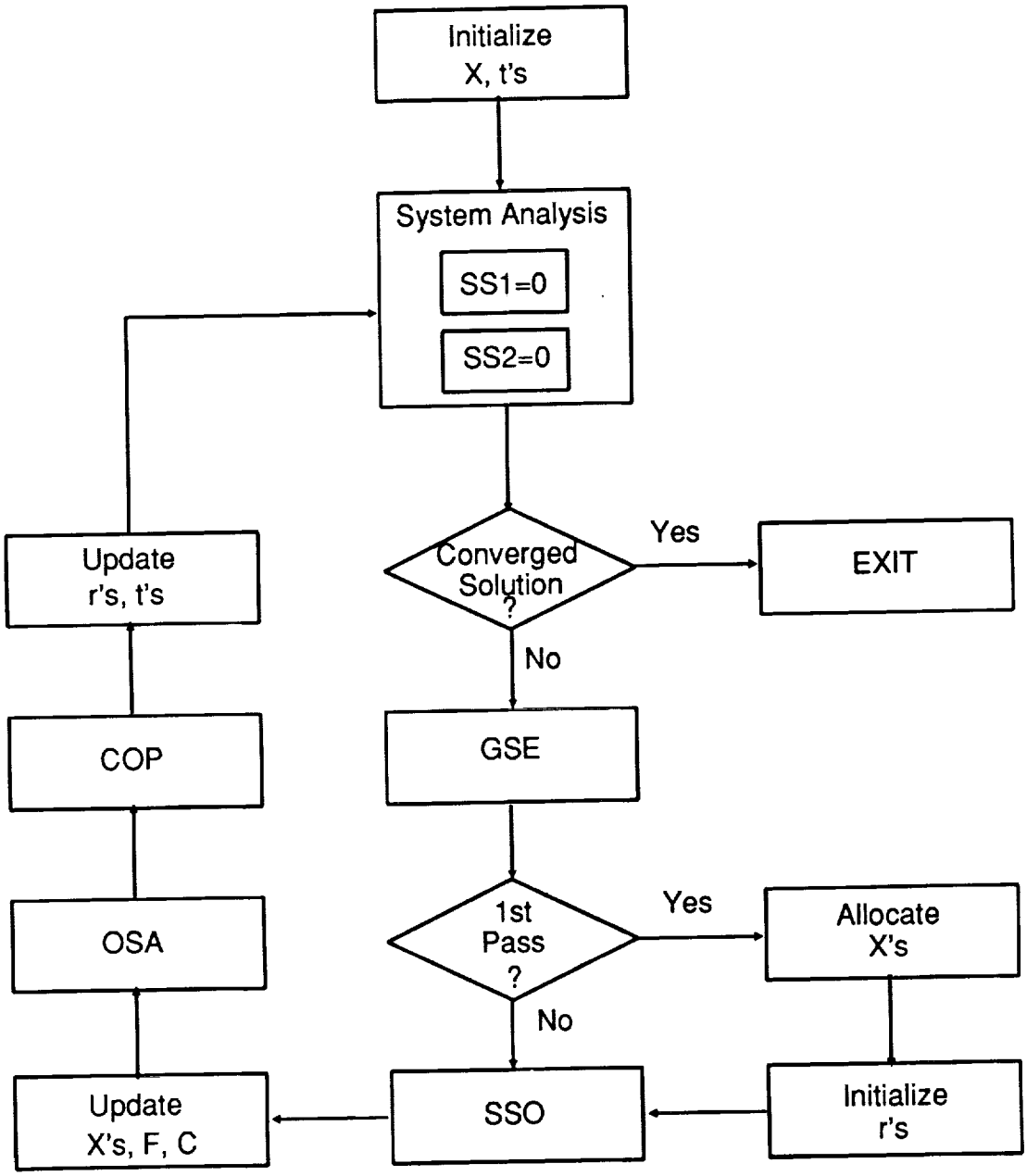


Figure 5.2 Flowchart for CSSO method.

of effectiveness coefficients. Effectiveness coefficients [Haj81], e_{ij}, essentially quantify the impact of a particular design variable, X_i, on the design at a point and can be written,

$$e_{ij} = \frac{dg_j / dX_i}{dF / dX_i} \tag{5.8}$$

where, g_j are the inequality constraints and F is the objective function. In order to determine the overall effectiveness of a design variable with respect to all design constraints simultaneously, it is necessary to rewrite Equation 5.8 in terms of a cumulative constraint. The effectiveness coefficients are now redefined in terms of only one subscript as,

$$e_i = \frac{dC / dX_i}{dF / dX_i} \tag{5.9}$$

Once effectiveness coefficients are determined for all subspaces, a rank-ordering procedure is used to determine the subspaces for which design variables have the greatest impact. For instance, in a two subspace system, effectiveness coefficients associated with each design variable and with the cumulative constraint for each of the two subspaces, would be determined. If a particular design variable is found to have a smaller effectiveness coefficient value (larger impact) for subspace 2 than for subspace 1, it is then allocated to subspace 2. Allocation of the design variables to the subspace upon which they have the greatest impact avoids potential divergence of the CSSO method.

Following design variable allocation, temporarily decoupled optimizations are performed in each subspace concurrently. The goal of these subspace optimizations (SSOs) is to reduce the violation of the cumulative constraint with the least increase of the system objective function or greatest decrease if the cumulative constraint is already satisfied. Essentially, the violated cumulative constraint is reduced only by some fraction,

with the other SSOs responsible for reducing the remainder of the violation. It is necessary that a feasible solution be obtained at the completion of the SSOs so that a constrained minimum exists. If such a result is not possible due to an initially infeasible design and restrictive move limits, constraint reduction techniques must be applied [Bar88].

The subspace optimization problem can be stated as follows,

$$\begin{aligned} &\text{Minimize} && F(X^k) \\ &\text{Subject to} && C^p \le C^{po}\left[s^p\left(1-r_k^p\right)+\left(1-s^p\right)t_k^p\right] && p=1,NSS \\ & && X_k^L \le X_k \le X_k^U && (5.10) \end{aligned}$$

where s^p, r^p_k, and t^p_k are coefficients representing cross influences of one subspace on another. Since the subspace optimizations are decoupled, with only subsets of the system design variables in each subspace, it is essential that some form of coordination exist between subspaces. The coordination coefficients perform this duty.

The r^p_k coefficient represents the 'responsibility' assigned to the kth SSO for reducing the violation of the cumulative constraint in the pth SSO. Even though design variables have been allocated to the subspace on which they have the greatest impact, it is easy to imagine how these variables would still have an effect on constraints of other subspaces due to the couplings which exist in the non-hierarchic system. It then becomes necessary to account for this effect during the subspace optimizations. The r^p_k coefficients essentially divide the responsibility for satisfying constraints amongst the subspaces according to the impact of the design variables within each subspace on the cumulative constraint.

The initialization of the r^p_k coefficients is based on the system sensitivities determined by the GSE. The sensitivity of the pth cumulative constraint with respect to the design variables associated with the kth subspace, is represented by the relation,

$$C_i^{pk} \equiv \frac{dC^p}{dX_i^k} \qquad i = 1, nxk \tag{5.11}$$

where, nxk is the number of design variables in the kth subspace. A matrix of cumulative constraint sensitivities is formed as follows.

$$J = \begin{array}{c} \\ k=1 \\ \vdots \\ \\ k=\xi \end{array} \begin{array}{c} \begin{array}{ccc} p=1 & \cdots & p=\xi \end{array} \\ \begin{bmatrix} C_1^{11} & & C_1^{\xi 1} \\ \vdots & \cdots & \vdots \\ C_{nx1}^{11} & & C_{nx1}^{\xi 1} \\ \vdots & \cdots & \vdots \\ C_1^{1\xi} & & C_1^{\xi\xi} \\ \vdots & \cdots & \vdots \\ C_{nx\xi}^{1\xi} & & C_{nx\xi}^{\xi\xi} \end{bmatrix} \end{array} \tag{5.12}$$

where, ξ is the total number of subspaces and nxξ is the number of design variables allocated to the ξth subspace. A variable, v^p_k, can be defined in terms of the maximum absolute value of sensitivities for each subspace. This variable corresponds to each column, p, of matrix J, as follows.

$$v_k^p = \frac{\max_i\left(\left|C_i^{pk}\right|\right)}{\max_k\left[\max_i\left(\left|C_i^{pk}\right|\right)\right]} \tag{5.13}$$

Scaling the v^p_k values such that the sum of the values over k for each column is unity, yields the r^p_k values as follows.

$$r_k^p = \left(\frac{1}{\sum_k v_k^p}\right) v_k^p \qquad p=1,\xi \tag{5.14}$$

Figure 5.3a demonstrates the responsibilities assigned to Subspaces 1, 2, and 3 for satisfying the cumulative constraint associated with Subspace 3. The responsibility coefficients for a particular 'p' constraint must equal 1.0 over all subspaces 'k'. This essentially means that 100% of the responsibility for satisfying a subspaces cumulative constraint must be accounted for.

The t^p_k coefficient represents the 'trade-off' associated with each subspace that allows for the violation of a constraint in the pth SSO in order to obtain a reduction of the objective function, provided that the constraint will be oversatisfied in the kth SSO. Such a trade-off can occur only when the present and previous optimization cycles have produced satisfied constraints. Figure 5.3b demonstrates the trade-off in constraint satisfaction which might occur amongst three subspaces. It is essential that any violation that is permitted be compensated in other subspaces so that the sum of all trade-offs across the subspaces is zero. The 'switch' parameter, s^p ,is responsible for enabling or disabling the r^p_k or t^p_k coefficients depending on whether the constraints are initially violated.

Following the subspace optimizations, a new constrained minimum point is defined. Due to the fact that the SSOs are formulated in terms of the coordination coefficients, r^p_k and t^p_k, the new optimal point is dependent on these variables. Therefore, it is possible to mathematically determine the sensitivity of the system objective function, F, to these variables by implementation of an optimum sensitivity analysis (OSA). Such an analysis is dependent upon Lagrange multipliers, which are defined in terms of the Kuhn-Tucker conditions [Van84].

At a constrained optimum, where X* defines the optimum design, the Kuhn-Tucker conditions require that,

$$\nabla \mathbf{F}(\mathbf{X}^*) + \sum_{j=1}^{m} \lambda_j \nabla \mathbf{g}_j(\mathbf{X}^*) + \sum_{k=1}^{l} \lambda_{k+m} \nabla \mathbf{h}_k(\mathbf{X}^*) = 0 \tag{5.15}$$

and

$$\lambda_j g_j(\mathbf{X}^*) = 0 \qquad j = 1, m \qquad \text{and} \qquad \lambda_j \geq 0 \tag{5.16}$$

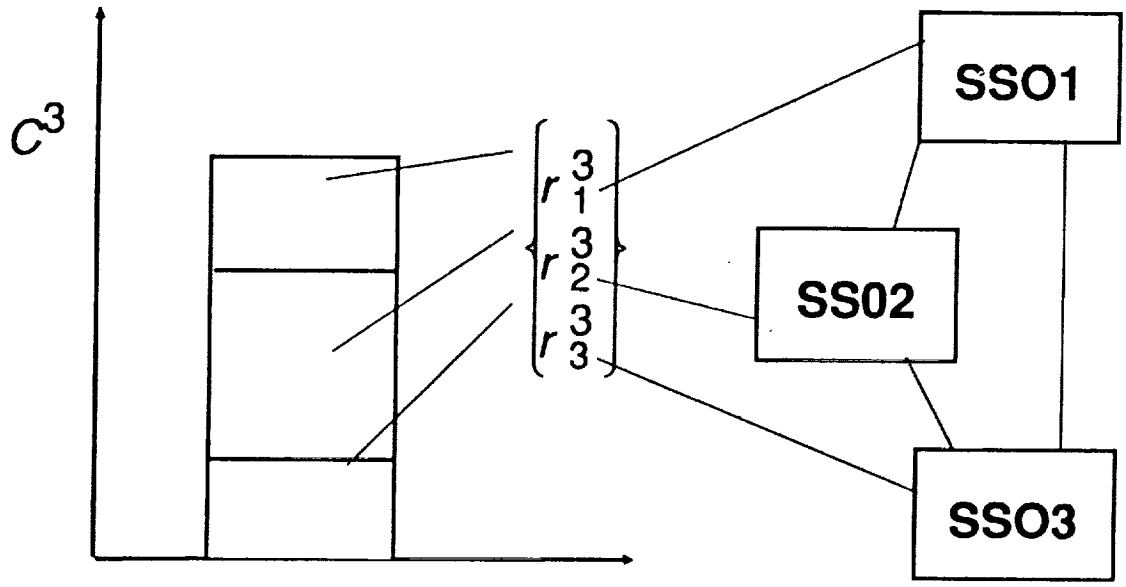


Figure 5.3a Distribution of values for responsibility coefficients in subspace three.

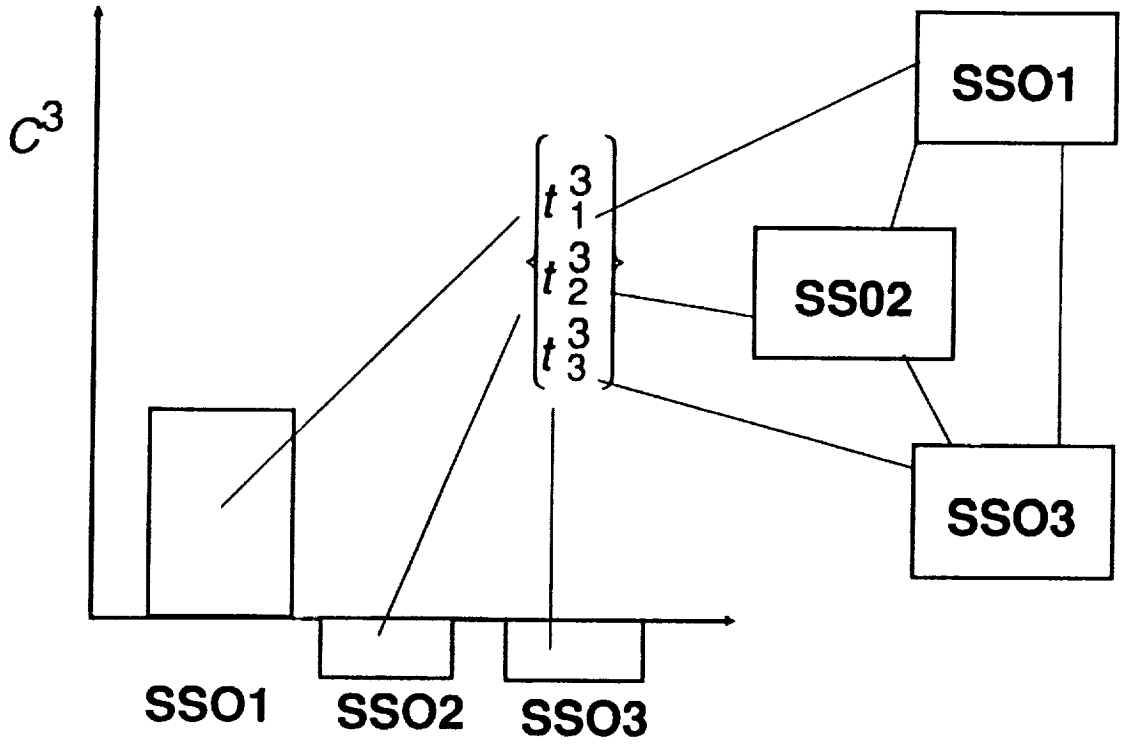


Figure 5.3b Trade-off coefficient values for subspace three.

where λ_j and λ_{k+m} are the Lagrange multipliers associated with the inequality and equality constraints, respectively. As can be seen from equation 5.15, only one Lagrange multiplier corresponds to each constraint. Equation 5.15 can be written for the case of no equality constraints and in terms of just one cumulative constraint, C, as follows,

$$\frac{dF}{dX_k^*} + \lambda_k \frac{dC}{dX_k^*} = 0 \tag{5.17}$$

where k represents the kth subspace. As previously described, in each subspace optimization, the system objective function is minimized with respect to a subset of the design variables and subject to constraints (defined in terms of the coordination coefficients) which are associated with each of the subspaces, thus yielding ξ constraints per SSO. Therefore, distinct values for F* are obtained in each subspace following the subspace optimizations.. Hence, Lagrange multipliers can be found corresponding not only to each constraint, but also to each subspace. This dictates a slightly different treatment of the Kuhn-Tucker conditions. Equation 5.17 can be rewritten to include consideration of each constraint in the SSO within the kth subspace as,

$$\frac{dF}{dX_k^*} + \sum_{p=1}^{\xi} \lambda_k^p \frac{dC^p}{dX_k^*} = 0 \tag{5.18}$$

Rewriting this expression in matrix form, it is possible to obtain a relation for the Lagrange multipliers as a function of constraint and objective function gradients associated with each subspace from Equation 5.18 as follows.

$$[\lambda_k] = -\left[\left[\frac{dC}{dX^k}\right]^T \left[\frac{dC}{dX^k}\right]\right]^{-1} \left[\frac{dC}{dX^k}\right]^T \left[\frac{dF}{dX^k}\right] \tag{5.19}$$

Here, $[\lambda_k]$ and $[dC/dX^k]$ are partitioned matrices of the form,

$$[\lambda_k] = [\lambda_k{}^1, \lambda_k{}^2, \ldots \lambda_k{}^\xi]^T \quad \text{and} \quad [dC/dX^k] = [dC^1/dX^k, dC^2/dX^k, \ldots dC^\xi/dX^k]^T \qquad (5.20)$$

Once the Lagrange multipliers are obtained, the optimum sensitivity of F simplifies to,

$$\frac{dF}{dz_i} = \sum_p \lambda_k^p \frac{dC^p}{dz_i} \qquad (5.21)$$

where z_i is a variable representing either r^p_k or t^p_k.

The derivative information obtained in the OSA is now used in the coordination optimization problem (COP) in which the system objective function is minimized with respect to the r^p_k and t^p_k coefficients. Completion of the COP yields new coefficients for use in the next SSO. The coordination optimization problem is defined in the following manner.

Minimize $\quad F(r_k^p, t_k^p)$

Subject to $\quad \sum_k r_k^p = 1$

$\sum_k t_k^p = 0$

$0 \le r_k^p \le 1$

$$r^p_{kL} \le r^p_k \le r^p_{kU} \quad \text{and} \quad t^p_{kL} \le t^p_k \le t^p_{kU} \qquad (5.22)$$

Following the update of the coefficients, the entire process is repeated until prescribed convergence requirements are met.

Certain advantages and disadvantages of the CSSO can be identified based on the performance of other decomposition-based algorithms. Due to linearizations which exist in both the SSO and the COP, move limits may be somewhat restrictive, depending on the

problem parameter sensitivity resulting from the OSA may be in error if active constraint switching occurs. The theorized merits of the approach, however, far outweigh the identified disadvantages. The most important feature of the CSSO is the fact that the modularity of the method allows for the efficient decoupling of the system to permit concurrent sensitivity analyses and subspace optimizations that can correspond to specialty groups within an organization. Further, the approach is particularly amenable to human judgement and intervention or the application of an artificial intelligence based expert system. These advantages and disadvantages are discussed in more detail in following chapters.

Heuristic Method: Concurrent Subspace Optimization - Embedded Expert System Method

A method is introduced that couples algorithmic and heuristic concepts to permit the 'intelligent' automated design of non-hierarchic systems. The method makes use of problem-dependent heuristics in the form of an embedded expert system capability. A development of this heuristic approach to system decomposition for design and optimization is presented.

Integration of the algorithmic aspects of the CSSO method with problem dependent heuristics is achieved with an embedded expert systems capability. The inference environment used is the 'C Language Integrated Production System' (CLIPS) [Gia89]. CLIPS is invoked in an embedded mode from within a FORTRAN program, thus providing a convenient link between procedural and heuristic processing of information.

CLIPS is an expert system shell comprised of three basic components - the fact-list, the knowledge base, and the inference engine. The facts (which are entered into a fact-list) in a CLIPS program are the data that stimulate the execution of the rules. They are entered into the fact-list with an assert command as follows

```
(assert    (fact1))
```

The fact within the inner set of parentheses can be composed of more than one field, with the first field quite often reserved to demonstrate a relation amongst the following fields, as in the example

```
(assert (constraint_value
                less_than_zero   equal_zero   greater_than_zero))
```

The defined rules use the asserted facts in the fact-list to make a program execute. The CLIPS format of these rules is analogous to an IF THEN statement in procedural languages. An example of such a pseudocode statement is

```
IF      the value of constraint is less than zero
THEN    the constraint is feasible.
```

The CLIPS format for this rule would be

```
(defrule constraint_status
        (assert (constraint_value       less_than_zero))
=>
        (assert (constraint_status      feasible)))
```

where the left hand side (LHS) of the rule contains the patterns (i.e. (constraint_value less_than_zero)) and the right hand side (RHS) contains the actions (i.e. (constraint_status feasible)). The rule is activated and put on the agenda if all the patterns of a rule match facts in the fact-list.

The knowledge-based problem solving system involves three basic levels of organization - the function, knowledge, and program levels [Ton87]. The function level corresponds to actual design implementation, the knowledge level contains detailed description of the design domain, and the program level contains the mechanics of implementing the design steps. Figure 5.5 demonstrates the organization of tasks with respect to these levels in the problem solving system previously described. The

implementation of these features is shown in the flowchart in Figure 5.6, in which the modified heuristics-based CSSO is shown. The specific applications of the embedded expert systems capability include the allocation of design variables among subspaces, the determination of optimization parameter values, the assignment of move limit values for efficient convergence, and the determination of coordination coefficient values. These tasks, which form the basis for the CSSO-EES method, are examined in more detail in Chapter 7.

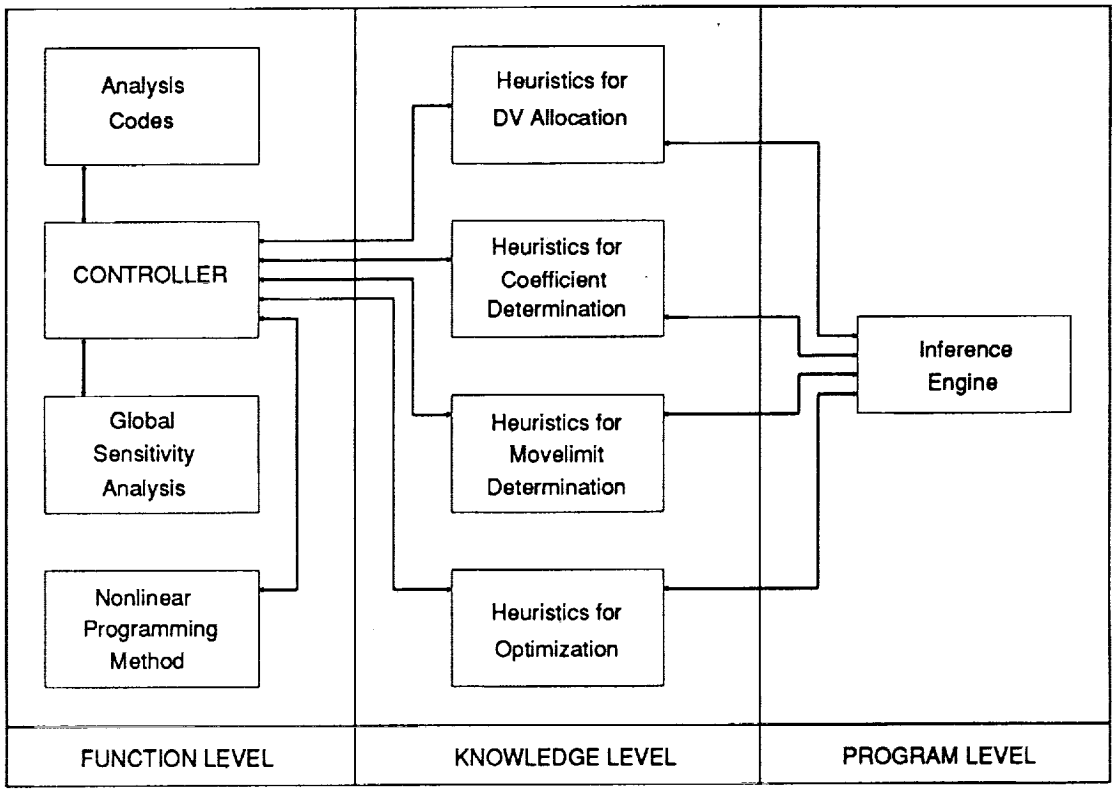


Figure 5.4 Organization of tasks in problem-solving system.

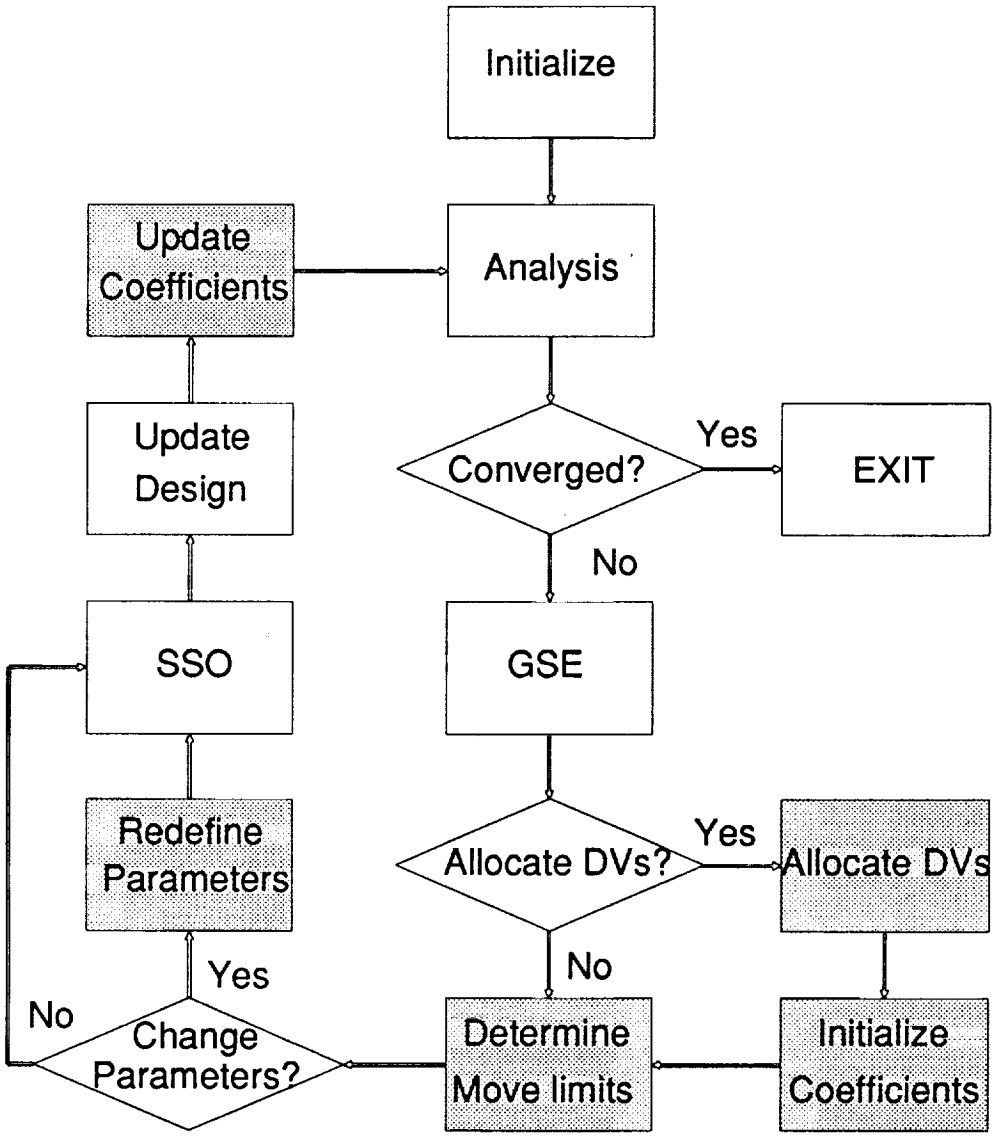


Figure 5.5 Flowchart for heuristics-based CSSO method.

CHAPTER 6
ANALYSIS METHODOLOGY AND MODEL DESCRIPTION

A description of the design objectives, application model, and analysis methodology is presented for each decomposition approach.

Global Sensitivity Equation Method

Design Objective

The objective of the synthesis process is to find the minimum weight configuration of a general aviation aircraft (Figure 6.1) subject to design limitations derived from structural integrity and aerodynamics/flight mechanics performance characteristics. Design variables contributing to the fulfillment of the optimization objectives stem from planform geometry and sizing of the aircraft. The non-hierarchic multidisciplinary environment of structures, aerodynamics, and flight mechanics is represented in terms of distinct analysis modules as seen in Figure 6.2. Each discipline module has inputs in the form of design variables that are intrinsic to that discipline as well as output data from other disciplines.

Application Model

The finite element analysis model for the structures discipline, with representative node and panel numbering, is shown in Figure 6.3. A stick model of the fuselage and tail structure represented by beam elements is connected to a built-up membrane/stringer model for the wing structure. A symmetric half of the structural model is used with a total of four hundred and twenty-six degrees-of-freedom. Definition of the aerodynamic model is in

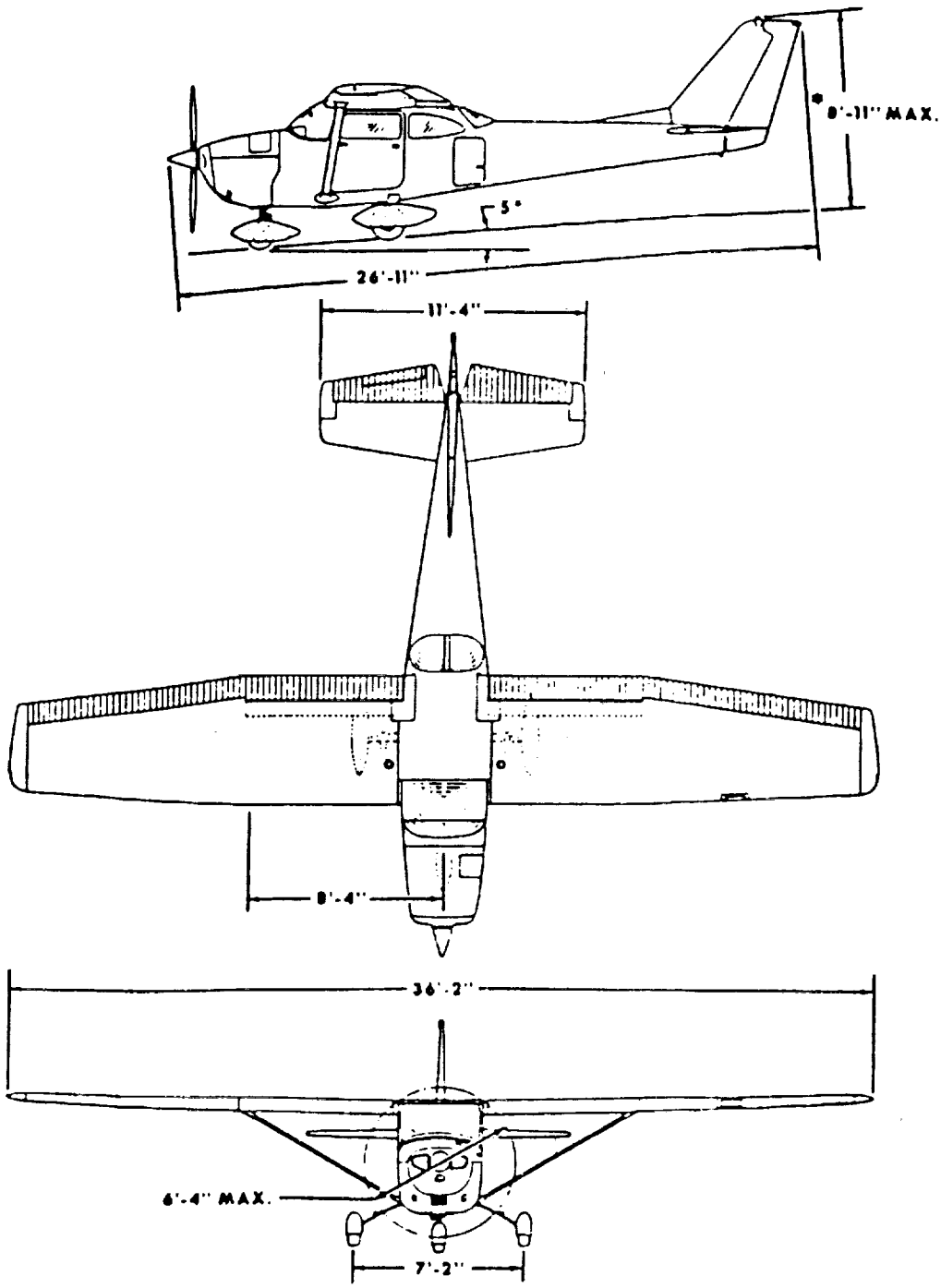


Figure 6.1 Three view of general aviation aircraft.

Source: J. Roskam, Airplane Flight Dynamics and Automatic Flight Controls, Pt. I (Roskam Aviation Engineering Corporation, Lawrence, Kansas, 1979, p. 590).

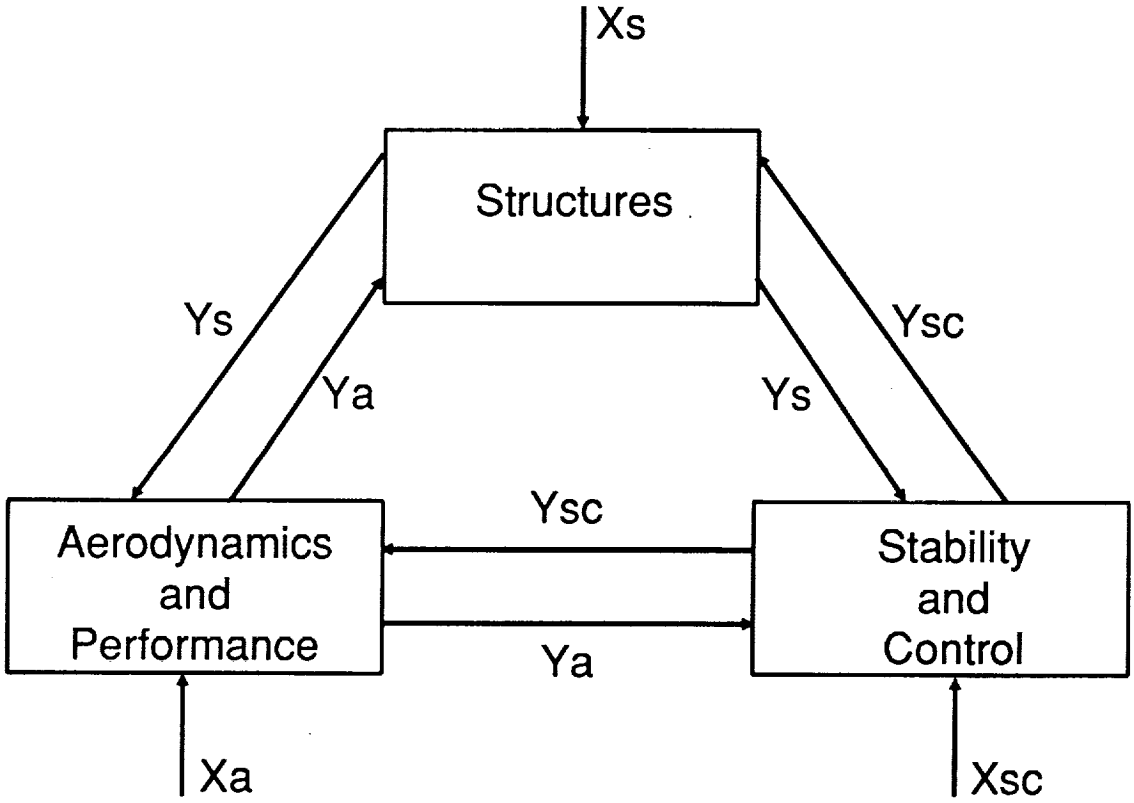


Figure 6.2 Subsystem interactions in multidisciplinary synthesis problem.

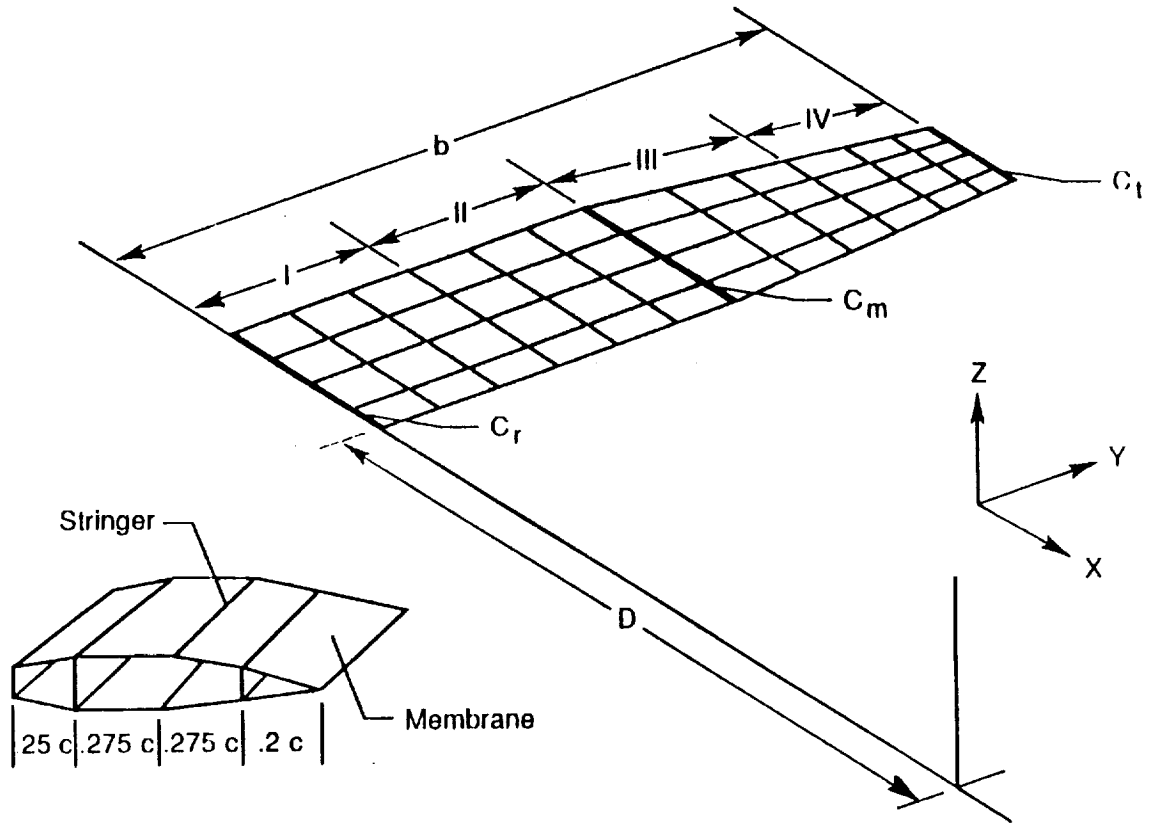


Figure 6.3 Structural finite element model.

accordance with the input requirements of an unsteady doublet lattice program ISAC [Pee79], and is shown in Figure 6.4. This figure also shows the discrete structural and aerodynamic analysis models used in this effort. A beam representation for the fuselage is retained. A lifting surface, defined as the aggregate of the upper and lower surfaces of a wing with a NACA 2412 airfoil, is modeled with plate elements. A similar approach is used to model the horizontal and vertical tails.

Analysis Methodology

Structures Subsystem. The equation for the free vibration eigenvalue problem associated with the structures subsystem is,

$$\left([K] - \omega_i^2 [M]\right)\{\phi\}_i = 0 \tag{6.1}$$

where $\{\phi\}_i$ and ω^2 are the eigenvector and eigenvalue for the ith mode, respectively. Generalized mass and stiffness matrices are defined as,

$$[\overline{M}] = [\phi]^T[M][\phi] \tag{6.2}$$

and

$$[\overline{K}] = [\phi]^T[K][\phi] \tag{6.3}$$

where the modes are normalized with respect to the mass matrix such that,

$$[\overline{M}] = [I] \tag{6.4}$$

The equations of equilibrium for linear static structural analysis are written as,

$$[K]\{u\} = \{P\} \tag{6.5}$$

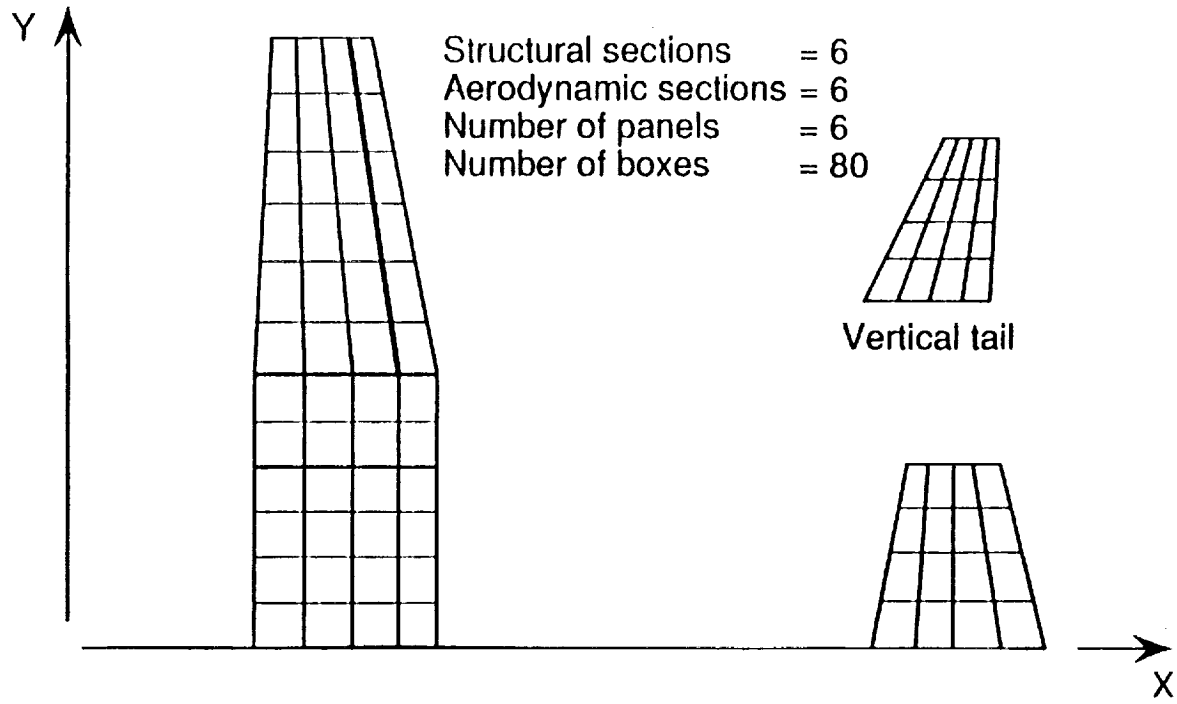


Figure 6.4 Aerodynamic model with panelling.

where {P} is the applied load vector. The element stresses are defined in terms of the nodal displacements as,

$$\{\sigma\} = [S]\{u\} \tag{6.6}$$

where [S] is the stress transformation matrix.

All structural analysis pertinent to the problem is performed using the finite element program 'Engineering Analysis Language' (EAL) [Whe83]. EAL is a high-order language with primary applications in analysis and design of solid and fluid systems based on a finite element representation of the analysis domain. Individual processors communicate through a data base containing information describing the finite element model of the structure, as well as data accumulated during execution of the runstream.

Aerodynamics and Performance Subsystem. In terms of primary structural coordinates, x, the equation of motion for a structure subject to aerodynamic loading can be written as follows.

$$\left[M_{\xi\xi}s^2 + (1+ig)M_{\xi\xi}\omega_\xi^2 + q_\infty Q_{\xi\xi}^M\right]\xi = q_\infty Q_\xi^G w_G \tag{6.7}$$

Here, the superscripts M and G denote quantities associated with the motion and gusts, respectively, and g is the structural damping coefficient. The gust time history is modeled as a deterministic sharp-edged gust. The generalized aerodynamic force matrix is defined as,

$$[Q(k,Ma)] = [\phi]^T[\Delta P(k,Ma)] \tag{6.8}$$

where k is the reduced frequency, Ma is the Mach number and ΔP is the differential pressure over the surface.

Performance requirements are defined as constraints in the optimization problem, and are determined from well documented relations [Per49]. The stall velocity is found from the relation,

$$V_s = \left[\frac{2W}{\rho_a C_{L_{max}} S} \right]^{1/2} \tag{6.9}$$

where W is the weight of the aircraft at take-off, ρ_a is the density of air at sea level, and S is the wing area. Another performance requirement is the landing distance over a fifty foot obstacle, and is calculated by summing the distance in the air D_A, and the distance on the ground D_G, where

$$D_A = \frac{W}{F} \left[\frac{V_{50}^2 - V_L^2}{2} + 50 \right] \tag{6.10}$$

and

$$D_G = -\frac{V_L^2}{2a} \tag{6.11}$$

The quantity (W / F) is the average resistance coefficient, a is the uniform deceleration on the ground, V_L and V_{50} are the velocity at landing and at a fifty foot height, measured in ft/sec. Similarly, the take-off distance over a fifty foot obstacle (D_{TO}) is found from,

$$D_{TO} = \frac{100\hat{K}}{9} - \frac{1000}{3} \tag{6.12}$$

where

$$\hat{K} = \frac{W^2}{SHP C_{L_{TO}} \Gamma} \tag{6.13}$$

and HP and Γ are the rated horsepower of the engine and the ratio of air densities, respectively, and C_{LTO} is the lift coefficient at take-off.

Breguet's Formula is used to determine the range. Based on the assumption that all fuel is used, the range R, in miles, can be expressed as,

$$R = 375 \frac{C_L}{C_D} \frac{N}{F} \ln\left(\frac{W}{W_p}\right) \tag{6.14}$$

where N is propeller efficiency factor, F is the specific fuel consumption, and W_p is the difference between the take-off weight and the weight of the fuel.

Stability and Control Subsystem. A first order state space representation is used for the analytical model of the flight mechanics subsystem. The equation of motion for a structure with active controls and subject to time varying airloads can be written in terms of airloads Q_i and modal displacements q_i as follows,

$$M_i \ddot{q}_i(t) + \omega_i^2 M_i q_i(t) + \sum_{j=1}^{n} Q_{ij} q_i(t) = -Q_{i\delta}\delta(t) - q_i(t) w_G(t) \tag{6.15}$$

where $\delta(t)$ is the control surface deflection and $w_G(t)$ is the gust velocity.

The dimensionality of the modal matrix is determined by the number of modes that are deemed necessary to model the structural displacements and other system degrees-of-freedom. Since the lower modes are dominant in representing the displacements, typically only the first six to ten modes are used in the analysis.

The first order state-space representation of the governing differential equations for the open-loop system can be written as follows,

$$\{\dot{x}\} = [A]\{x\} + [B]\{u\} + [H]\{n\} \tag{6.16}$$

where A is the system dynamics matrix, B and H are the control and gust distribution matrices, respectively; x is the state variable vector; u is the control input vector, and n is the gust vector.

In terms of the state vector x, the system output y can be written as,

$$\{y\} = [C]\{x\} \tag{6.17}$$

where the [C] matrix contains information specific to the location of the sensors. The output vector is of length s, where s is the number of sensors present in the system.

The optimal state feedback control law can then be found as a function of the gain matrix as follows.

$$\{u\} = -[G]\{x\} \tag{6.18}$$

The use of this relationship in Equation 6.16 yields an expression which allows the determination of the optimal state for the closed-loop system. A time-marching method is used to determine the time history for the state variables. Once the state solution is known, the dynamic displacements can then be retrieved for each degree-of-freedom. The control input resulting from this analysis is used to determine the mass of the physical control system. This mass is used as an input to the structural system and directly influences the structural dynamic characteristics.

The required stability derivatives for the stability and control analysis are obtained through a semi-elastic stability analysis, which constitutes a modification of rigid body stability characteristics to account for structural deformations. The relationship,

$$[\overline{K}]\{q\}+q_{\infty}[Q]\{q\}=0 \tag{6.19}$$

between the stiffness matrix and generalized aerodynamic force matrix, [Q], yields an elastic generalized aerodynamic force vector, $\{\bar{Q}\}$, as a function of rigid and elastic terms such that

$$\begin{Bmatrix}\bar{Q}_{12}\\ \bar{Q}_{22}\end{Bmatrix}=\begin{Bmatrix}Q_{12}\\ Q_{22}\end{Bmatrix}-q_\infty\begin{bmatrix}Q_{13} & \cdots & Q_{1n}\\ Q_{23} & \cdots & Q_{2n}\end{bmatrix}\left[\bar{K}+q_\infty Q\right]^{-1}\begin{Bmatrix}Q_{32}\\ Q_{n2}\end{Bmatrix} \tag{6.20}$$

where n is the number of elastic and rigid body modes and the first two rows of [Q] correspond to pitch and plunge.

Selected stability derivatives are determined in terms of the semi-elastic generalized aerodynamic force vector which are then used in the stability and control analysis to obtain the eigenvalues of the characteristic equation. The time-to-half is determined from the relation [Etk82],

$$t_{1/2}=\frac{0.69\,t^*}{\zeta} \tag{6.21}$$

where ζ is the eigenvalue for the mode under consideration and,

$$t^*=\frac{c}{2U} \tag{6.22}$$

defined in terms of the mean chord length c and the velocity U.

The rigid-body stability and control analysis is performed using a modified version of programs available in [Sme84].

Concurrent Subspace Optimization Method

Design Objective

The objective of the synthesis process is to find the minimum weight design of a truss structure subject to constraints derived from requirements of structural strength and

stiffness. Design variables contributing to the fulfillment of the optimization objectives are structural element sizing variables and a structural geometry variable.

The non-hierarchic environment which exists in the size/topology system is represented in terms of distinct analysis modules as shown in Figure 6.5. Each subsystem module has inputs in the form of design variables that are intrinsic to that subsystem as well as output data from the other subsystem.

Application Model

The ten-bar truss structure in Figure 6.6 was used to demonstrate the feasibility of the CSSO method. Two degrees of freedom (x and y) are permitted at each of the four unconstrained nodes, thus yielding an eight degree-of-freedom system. The structure is subjected to static loadings as shown in the figure.

Analysis Methodology

Subsystem 1 - Sizing The design variables for Subsystem 1 are the cross-sectional areas of the ten truss members. The output vector for the analysis associated with the subsystem contains the sizing variables, the objective function value, and a cumulative constraint measure representing the static stress constraints. The vectors are written:

$$\{X_{SS1}\}^T=\{A_1,...,A_{10}\} \tag{6.23}$$

and

$$\{Y_{SS1}\}^T=\{A_1,...,A_{10},W,C_{SS1}\} \tag{6.24}$$

Subsystem 2 - Topology The design variable for the topology subsystem is a geometry variable, D, which controls the depth of the truss structure at the wall. The

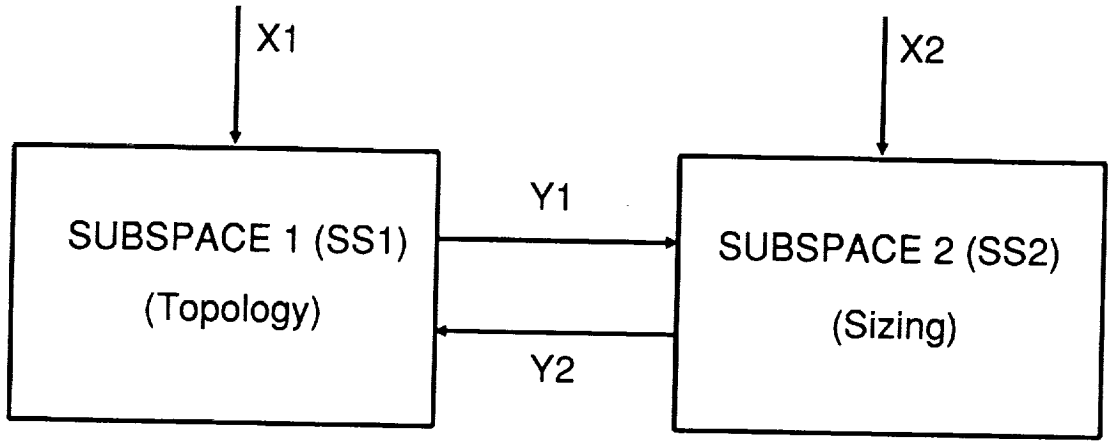


Figure 6.5 Subsystem interactions in size/topology problem.

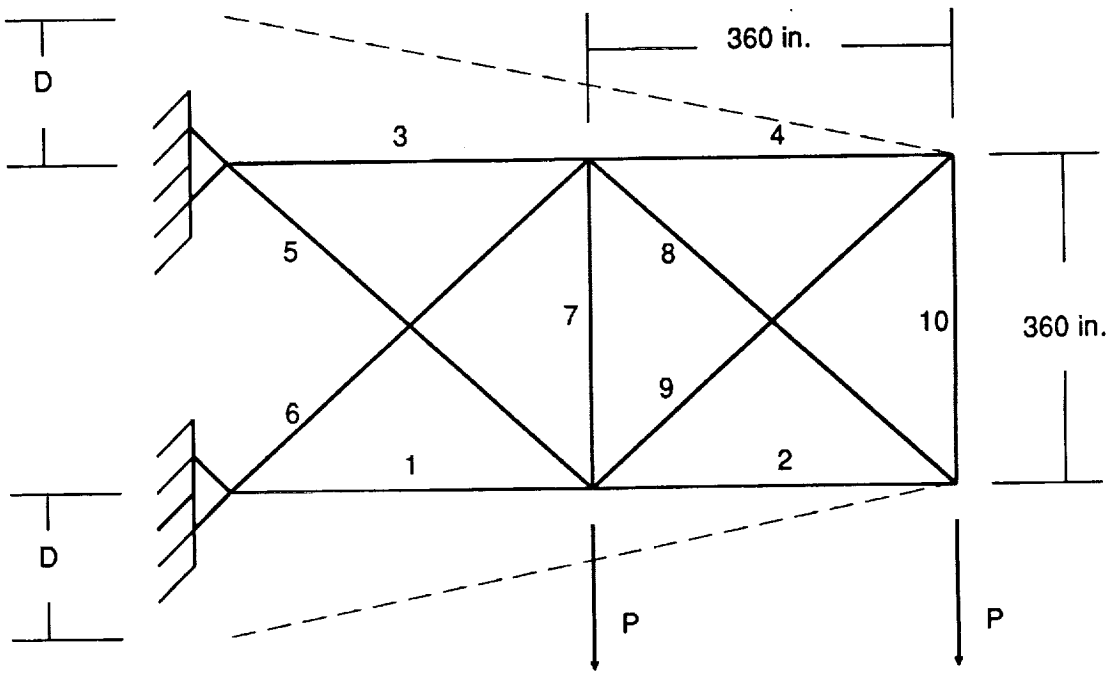


Figure 6.6 Structural truss model for CSSO verification.

output vector from the analysis contains the geometry variable, the objective function value, and a cumulative constraint measure representing the static lateral displacement constraints. The vectors associated with the subsystem are written:

$$\{X_{SS2}\}^T=\{D\} \tag{6.25}$$

and

$$\{Y_{SS2}\}^T=\{D,W,C_{SS2}\} \tag{6.26}$$

Inclusion of the design variables in the output vectors associated with both subspaces is a special case. Here, the analysis of one subspace requires the design variables of the other subspace as input variables, thus establishing a coupling between subspaces. All structural analysis pertinent to the problem was performed using the finite element program EAL.

Concurrent Subspace Optimization - Embedded Expert System Method

Design Objective

The design objective of the control/structures interaction (CSI) problem is to find the minimum weight cantilever ten-bar truss structure subject to constraints on static stresses, natural frequencies, and static and dynamic displacements. Design variables are contributed from both disciplines and include truss member sizing variables and a controls damping constant. Figure 6.7 demonstrates the subsystem coupling which exists in this problem.

Application Model

The ten-bar truss structure in Figure 6.8 is used to demonstrate the effectiveness of the CSSO-EES methodology. The truss is equipped with active controls to limit the dynamic displacements to preassigned levels. Two degrees-of-freedom (x and y) are

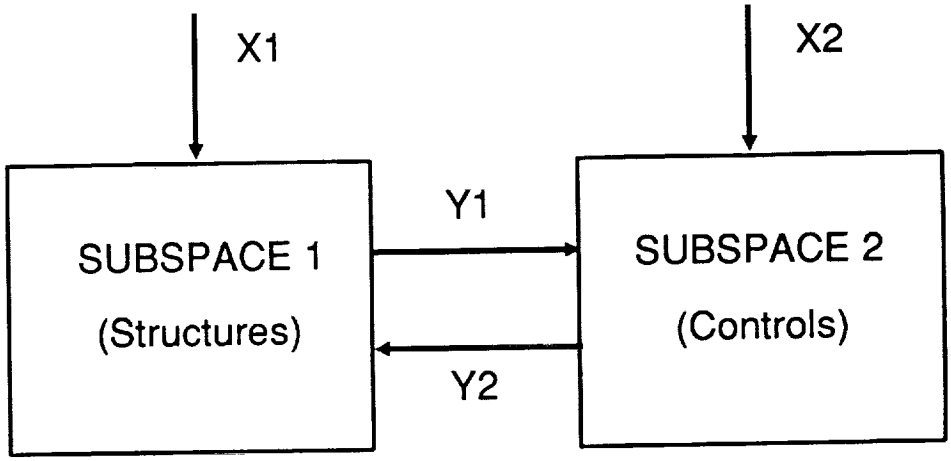


Figure 6.7 Subsystem interactions in CSI problem.

permitted at each of the four unconstrained nodes, thus yielding an eight degree-of-freedom system.

The structure is subjected to static and dynamic loadings as shown in Figure 6.9. The lateral dynamic displacements are controlled by four sets of hydraulic actuators placed at the unconstrained nodes of the truss. The forcing function, f(t), is a ramp input applied over a two second interval.

Analysis Methodology

Subsystem 1 - Structures The governing equation for the free vibration eigenvalue problem associated with the structures subsystem is

$$\left([K]-\omega_i^2[M]\right)\{\phi\}_i=0 \tag{6.27}$$

where $\{\phi\}_i$ and ω_i^2 are the eigenvector and eigenvalue for the ith mode, respectively and the structural eigenvalue analysis is obtained from the finite element program EAL.

The design variables associated with the structures analysis are the cross-sectional areas of the ten truss members. The output vector for the analysis associated with Subsytem 1 contains the eigenvector and eigenvalue information, the structural weight, and a cumulative constraint measure representing the frequency, static stress, and static displacement constraints. The vectors are written:

$$\{X_{SS1}\}^T=\{A_1,...,A_{10}\} \tag{6.28}$$

and

$$\{Y_{SS1}\}^T=\left\{\omega^2,\phi,W_S,C_{SS1}\right\} \tag{6.29}$$

Subspace 2 - Controls The equation of motion for an actively controlled structure subjected to forced vibration can be written,

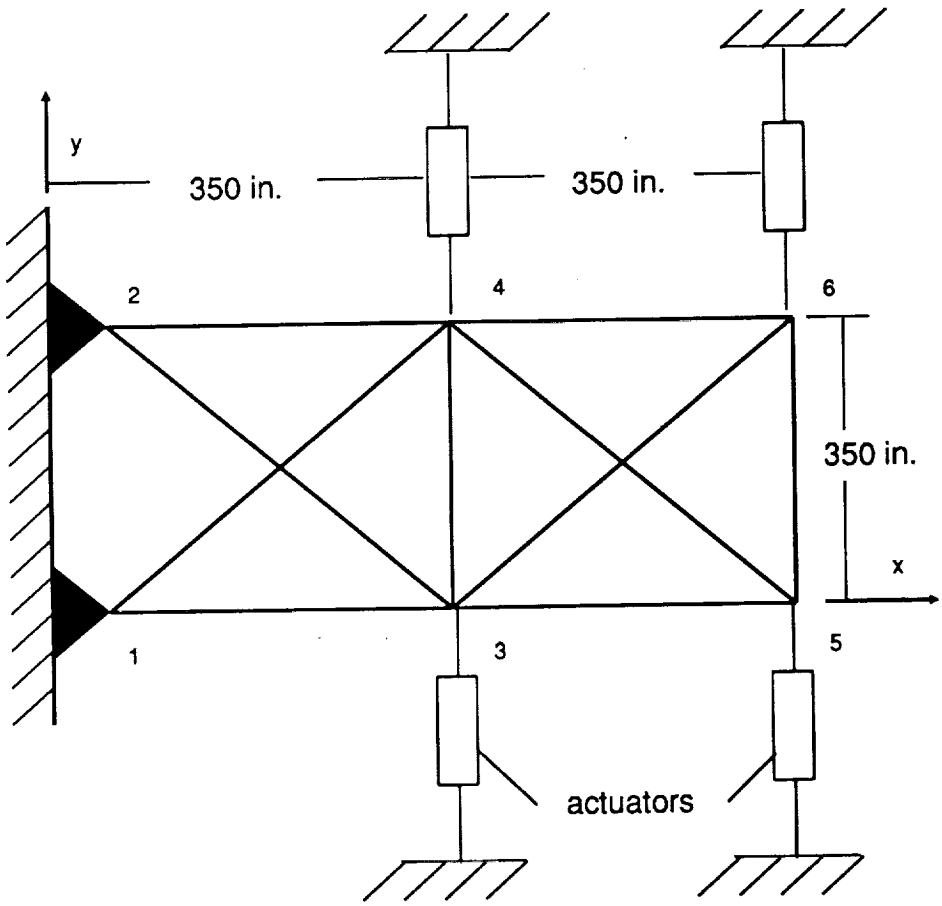


Figure 6.8 Cantilever truss with active controls.

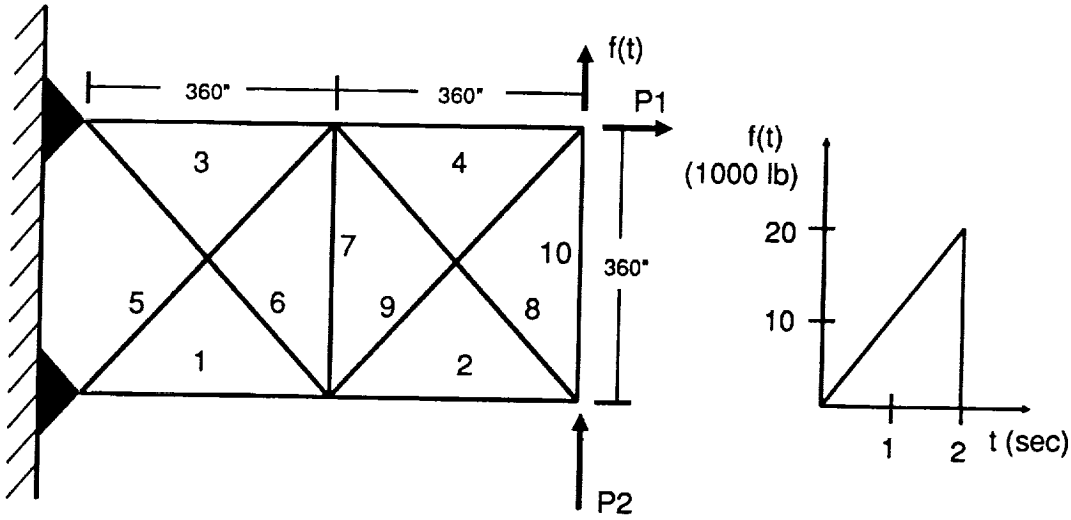


Figure 6.9 Ten-bar truss with static and dynamic loading.

$$[M]\{\ddot{r}\}+[C]\{\dot{r}\}+[K]\{r\}=[b]\{u\}+[\bar{b}]\{f\} \tag{6.30}$$

where [b] is a matrix containing information concerning location of actuators, $[\bar{b}]$ is a matrix containing applied load information, {r} is the displacement vector, {u} is control input, and {f} is the dynamic forcing function. The damping matrix in Equation 6.30 is traditionally represented by a proportionality relationship as follows,

$$[C] = \alpha\,[K] + \beta\,[M] \tag{6.31}$$

where α and β are proportionality constants.

The first-order state-space representation of the governing differential equations for the open-loop system can be written as,

$$\{\dot{x}\}=[A]\{x\}+[B]\{u\}+[\bar{B}]\{f\} \tag{6.32}$$

where {x}, {u}, and {f} are the state, control input, and forcing function vectors, respectively, and [A], [B], and $[\bar{B}]$ are the plant, control, and forcing matrices. The state vectors are defined in terms of the dynamic displacement, velocity and acceleration vectors as follows.

$$\{\dot{x}\}=\begin{Bmatrix}\dot{r}\\ \ddot{r}\end{Bmatrix} \qquad \text{and} \qquad \{x\}=\begin{Bmatrix}r\\ \dot{r}\end{Bmatrix} \tag{6.33}$$

A modal reduction technique is applied in which a modal transformation is made of the form,

$$\{r\} = [\phi]\{\eta\} \tag{6.34}$$

where $\{\eta\}$ is the transformed displacement vector. The above transformation can be applied to equation 6.30 to obtain,

$$[\overline{M}]\{\ddot{\eta}\}+[\overline{C}]\{\dot{\eta}\}+[\overline{K}]\{\eta\}=[\phi]^T[b]\{u\}+[\phi]^T[\overline{b}]\{f\} \tag{6.35}$$

where

$$[\overline{M}]=[I] \qquad [\overline{C}]=[2\zeta\omega] \qquad [\overline{K}]=[\omega^2] \tag{6.36}$$

all of which are diagonal matrices.

In modern control theory, the control vector {u} is determined using a linear quadratic regulator. The optimal state feedback control law is determined by minimizing a quadratic performance index [Bry69] which is a function of the state and control vectors such that,

$$PI=\int_0^\infty \left(\{x\}^T[Q]\{x\}+\{u\}^T[R]\{u\}\right)dt \tag{6.37}$$

where [Q] and [R] are arbitrary weighting matrices. The solution of the optimal control problem yields the nonlinear algebraic Riccati equation [Bry69] as follows.

$$[A]^T[P]-[P][B][R]^{-1}[B]^T[P]+[P][A]+[Q]=0 \tag{6.38}$$

The control gain matrix is defined in terms of the Riccati matrix, [P], the positive definite solution to Equation 6.38, as,

$$[G]=[R]^{-1}[B]^T[P] \tag{6.39}$$

The optimal state feedback control law can then be formulated in terms of the gain matrix to yield the optimal control input such that,

$$\{u\} = -[G]\{x\} \tag{6.40}$$

The optimal state can now be determined by a time-marching method to yield the dynamic displacements. The mass of the control hardware is expressed as an explicit function of the control input. The controls analysis was performed through implementation of a package of FORTRAN subroutines named 'Optimal Regulator Algorithms for the Control of Linear Systems' (ORACLS) [Arm78].

The design variable for the controls subsystem was a damping variable, c, defined as

$$c=\frac{2\zeta_i}{\omega_i} \tag{6.41}$$

where ζ_i is a damping coefficient. The output vector from the analysis contains the mass of the controllers, the weight of the control system, and a cumulative constraint measure representing the dynamic lateral displacement constraints. The vectors associated with the subsystem are written:

$$\{X_{SS2}\}^T = \{c\} \tag{6.42}$$

and

$$\{Y_{SS2}\}^T = \{m_c, W_c, C_{SS2}\} \tag{6.43}$$

CHAPTER 7
IMPLEMENTATION OF SOLUTION TECHNIQUES

The implementation of solution techniques for the three system decomposition method are presented. The GSE applications center on strategies to increase efficiency and solution accuracy for large problems. The CSSO and its heuristic counterpart, the CSSO-EES, are evaluated in terms of verification procedures to determine their feasibility.

Global Sensitivity Equation Method

In the use of the GSE method for the design problem defined in the previous chapter, the dimensionality of the global sensitivity matrix is of some concern. If each subsystem represents a discipline in a multidisciplinary optimization problem, it is conceivable that for a large number of outputs associated with each discipline, the dimensionality of global sensitivity matrix can be potentially quite large. The system of linear algebraic equations that are obtained by application of the GSE method can be expressed as,

$$[D]\{w\}=\{v\} \tag{7.1}$$

where [D] is the GSE partitioned matrix (GSM) containing the local sensitivity information of each subsystem, {v} is a known column vector of partial sensitivities, and {w} is the unknown column vector of total derivatives. If the vector {w}, required in forming the response approximations for each piecewise linear representation of the system, is obtained by decomposition of [D], the process can become unacceptably expensive.

Iterative Solution Technique

The present work adopts an alternative iterative solution to this system of equations, where an initial approximation to the solution is assumed and is successively modified to a converged solution. In this investigation, Gauss-Siedel iteration with relaxation is implemented to encourage convergence.

Gauss-Siedel iteration is recursive in nature, as one repeatedly cycles through solutions for the unknowns, which then replace the old values. The method, thus, automatically makes use of the most recently calculated values for the unknowns, resulting in large computer storage savings, as only one value for each unknown need be saved. A point relaxation technique is implemented, wherein the calculated value of the unknown is modified as,

$$w_i^{(m+1)} = w_i^{(m)} + \lambda\left(w_i^{(m+1)^*} - w_i^{(m)}\right) \tag{7.2}$$

where λ is the relaxation factor, (m+1) is the current iteration, and $w_i^{(m+1)^*}$ is the value for the unknown obtained by the Gauss-Siedel approach in the current iteration.

System Conditioning Evaluation

The level of ill-conditioning associated with matrix [D] is expressed in terms of a condition number [Don77] which is defined as,

$$\| D \|_1 \| D^{-1} \|_1 \tag{7.3}$$

Here, the first norm of [D] is used and has the form,

$$\|D\|_1 = \max_j \sum_i |d_{ij}| \tag{7.4}$$

The quantity $\| D^{-1} \|_1$ is estimated by the relation,

$$\left\| D^{-1} \right\|_1 = \frac{\|z\|}{\|y\|} \tag{7.5}$$

where vector {y} is chosen and {z} is then determined from the system of equations,

$$[D]\{z\}=\{y\} \tag{7.6}$$

System Normalization Requirements

The condition number is a measure by which the accuracy of the solution may be gaged and is determined by the relative magnitude of the terms in the GSE matrix (GSM). Since the components of the output response vector Y and the design variable vector X are of varying magnitudes, it is necessary to scale the partial sensitivity terms in the GSM. A normalization scheme was implemented to achieve this, and is most easily described by considering two systems 1 and 2, with scalar intrinsic design variables and scalar output responses. To determine the total derivatives of the outputs with respect to the design variable of subsystem 1, the GSE can be written in matrix form as follows.

$$\begin{bmatrix} 1 & -\frac{\partial Y_1}{\partial Y_2} \\ -\frac{\partial Y_1}{\partial Y_2} & 1 \end{bmatrix} \begin{Bmatrix} \frac{dY_1}{dX_1} \\ \frac{dY_2}{dX_1} \end{Bmatrix} = \begin{Bmatrix} \frac{\partial Y_1}{\partial X_1} \\ 0 \end{Bmatrix} \tag{7.7}$$

The partial sensitivities on the left and right hand sides of the equation are normalized as,

$$\frac{\partial Y_1}{\partial Y_2}^* = \frac{Y_2}{Y_1}\frac{\partial Y_1}{\partial Y_2} \qquad \frac{\partial Y_2}{\partial Y_1}^* = \frac{Y_1}{Y_2}\frac{\partial Y_2}{\partial Y_1} \qquad \frac{\partial Y_1}{\partial X_1}^* = \frac{X_1}{Y_1}\frac{\partial Y_1}{\partial X_1} \tag{7.8}$$

yielding the normalized Global Sensitivity Equations.

$$\begin{bmatrix} 1 & -\frac{\partial Y_1}{\partial Y_2}^* \\ -\frac{\partial Y_1}{\partial Y_2}^* & 1 \end{bmatrix} \begin{Bmatrix} \frac{dY_1}{dX_1}^* \\ \frac{dY_2}{dX_1}^* \end{Bmatrix} = \begin{Bmatrix} \frac{\partial Y_1}{\partial X_1}^* \\ 0 \end{Bmatrix} \tag{7.9}$$

The unscaled behavior derivatives are then recovered from the scaled values by the relationship,

$$\frac{dY_1}{dX_1} = \frac{Y_1}{X_1}\frac{dY_1}{dX_1}^* \qquad \frac{dY_2}{dX_1} = \frac{Y_2}{X_1}\frac{dY_2}{dX_1}^* \tag{7.10}$$

Solution Standard Deviation Comparison

The behavior sensitivities obtained from an application of the GSE approach, using both direct decomposition and iterative methods, were compared with results obtained from a forward finite difference approach applied to the coupled system. The percent difference in the two solutions under comparison is written as,

$$Per_{i,k} = 100 \times \frac{ABS\left[\frac{dY_i}{dX_{k_1}} - \frac{dY_i}{dX_{k_2}}\right]}{MAX\left[\frac{dY_i}{dX_{k_1}}, \frac{dY_i}{dX_{k_2}}\right]} \quad i=1,NGSM \text{ and } k=1,NX \tag{7.11}$$

where NGSM is the dimensionality of the GSM. A variance of $Per_{i,k}$ [Pre86] is then defined as follows.

$$Var(Per_{i,k}) = \frac{1}{NGSM}\sum_{i=1}^{NGSM}\left(Per_{i,k}\right)^2 \tag{7.12}$$

A standard deviation measure of the variance is adopted for convenience and is defined as follows.

$$\bar{\alpha}(\mathrm{Per}_{i,k}) = \left[\mathrm{Var}\left(\mathrm{Per}_{i,k}\right)\right]^{1/2} \tag{7.13}$$

Constraint Reduction Implementations

Constraints for the multidisciplinary synthesis problem described in the previous chapter are as follows. Structural constraints were placed on the first and second natural frequencies, on the eight lateral wing tip displacements, on the stress constraints for the root section stringers and membranes, and on the internal volume of the wing box. Aerodynamic performance constraints were placed on stall velocity, landing and take-off distances over a fifty foot obstacle, and range. Controls constraints were placed on the dynamic lateral displacements of the wing and horizontal tail, the deflection of the control surface, and the time-to-half for the two longitudinal modes.

The iterative solution to the global sensitivity equations was implemented in conjunction with an approach to reduce the dimensionality of the system equations. Explicitly, this involves the reduction in the total number of subsystem output parameters by an efficient constraint representation approach. Such an approach permits the representation of a large number of inequality constraints by a single cumulative measure in the form of a K.S. function.

Solutions of the global sensitivity equations were obtained for three specific cases, with selective use of the cumulative constraint to reduce the system dimensionality.

Case 1. Constraint reduction techniques are not used. The system output vectors for the five mode case are as follows.

$$
\begin{aligned}
Y_S &= \left(\omega^2, \phi, \overline{K}, V, I_{yy}, W, g_S\right) && NS = 453 \\
Y_A &= \left(L, C_{L_\alpha}, C_{M_\alpha}, g_A\right) && NA = 30 \\
Y_C &= \left(m_C, \delta, g_C\right) && NC = 19
\end{aligned} \tag{7.14}
$$

The GSM dimensionality for this case is 502 x 502.

Case 2. Cumulative constraints are used for static stresses and for static and dynamic displacements, resulting in output vectors with the following dimensions for the five mode case.

$$
\begin{aligned}
NS &= 400 \\
NA &= 30 \\
NSC &= 5
\end{aligned} \tag{7.15}
$$

The GSM has dimensions 435 x 435.

Case 3. Cumulative constraints are used to represent all constraints in each subsystem resulting in output vector dimensions as follows.

$$
\begin{aligned}
NS &= 389 \\
NA &= 25 \\
NSC &= 3
\end{aligned} \tag{7.16}
$$

The GSM has dimensions 417 x 417 for this case.

Design Variable Allocation Comparisons

The six design variables that may participate as local variables to each subsystem are

$$\{Cr, Cm, Ct, b, \gamma, D\} \tag{7.17}$$

The variables Cr, Cm, Ct correspond to the chord lengths at the root, mid-station, and tip of the wing. The variables b and γ correspond to the semi-span and dihedral of the wing. The placement of the wing on the fuselage is determined by D which represents the distance between the horizontal tail quarter-chord point and the trailing edge of the wing at the root section.

Additional design variables traditionally allocated to the structures subsystem correspond to stringer and membrane thicknesses in the four prescribed wing sections and were as follows.

$$\{a_1,...,a_8,Th_1,...,Th_{12}\} \tag{7.18}$$

Here, a_1 and a_2 correspond to the bottom and top stringer areas of section I, a_3 and a_4 correspond to the bottom and top stringer areas of section II, etc. Similarly, Th_1 , Th_2 , and Th_3 correspond to the bottom, top, and side membrane thicknesses of section I.

The gain components of the optimal control analysis were considered to be design variables in this multidisciplinary synthesis problem and are traditionally placed in the flight mechanics subsystem. The gain matrix, G, has dimensions nac x 2nmod where nac is the number of actuators and 2nmod is twice the number of modes considered in the analysis.

$$(G_1,...,G_{2nmod}) \tag{7.19}$$

Since certain design variables affect the analyses, and hence the design constraints for more than one subsystem, these design variables can be represented in more than one way in the GSM. Two representations were implemented in this study.

Case 1. In this case, design variables which contribute to more than one subsystem analysis were considered design variables in each of the subsystems. The design variable vectors are,

$$X_S = (C, b, \gamma, D, a, Th) \qquad NXS = 26$$
$$X_A = (C, b, \gamma, D) \qquad NXA = 6$$
$$X_S = (C, b, \gamma, D, G) \qquad NXC = 8, 12, 16 \qquad (7.20)$$

where C are chord lengths at designated span stations, b is the wing semi-span, γ is the wing dihedral, D defines the wing location along the fuselage, a are the stringer areas, Th are membrane thicknesses, and G is the controller gain vector which has dimensions of two times the number of eigenmodes used in the analysis. As stated previously, one, three, and five modes were used in the numerical work.

Case 2. In this implementation, specific design variables were allocated to only one discipline, but were also represented as output in the Y vectors so their influence was still retained in other subsystem analyses. The design variable vectors are designated as follows.

$$X_S = (a, Th) \qquad NXS = 20$$
$$X_A = (C, b) \qquad NXA = 4$$
$$X_S = (\gamma, D, G) \qquad NXC = 4, 8, 12 \qquad (7.21)$$

Here, the choice of design variables was critical, as it affects the conditioning of the GSE system matrices.

Concurrent Subspace Optimization Method

Verification Procedure

The validity of the CSSO method is established by application to the structural synthesis problem defined in the previous chapter. The design variables are permanently

assigned to the subspaces at the outset of the design procedure in order to simplify the initial method application. Optimization results for this structural problem are obtained by three different synthesis procedures for comparison purposes, one of which is the CSSO Method. An all-in-one optimization is performed for the other two cases in which the sensitivity information is obtained by the finite difference method and by the GSE method.

Distributed Processing Environment

The advantage of modularity is investigated by implementation of a distributed processing scheme in order to demonstrate the versatility and potential computational efficiency of the CSSO method. Subsystem analyses are performed in separate computing environments concurrently in order to achieve a parallel processing capability. The schematic implementation of this capability is shown in Figure 7.1.

Approximation Scheme Comparison

The application of various approximation schemes for constraint representations including linear, reciprocal, and improved [Fad90] approximations were investigated in this effort. A comparison of optimization convergence histories and constraint violation histories were made to determine the most effective scheme for the class of problems considered.

Coefficient Effect Evaluation

The effect of the r and t coefficients on the convergence characteristics for the optimization process was investigated. A study is made to determine whether bounding the t coefficients yield superior convergence characteristics as well as reduced constraint

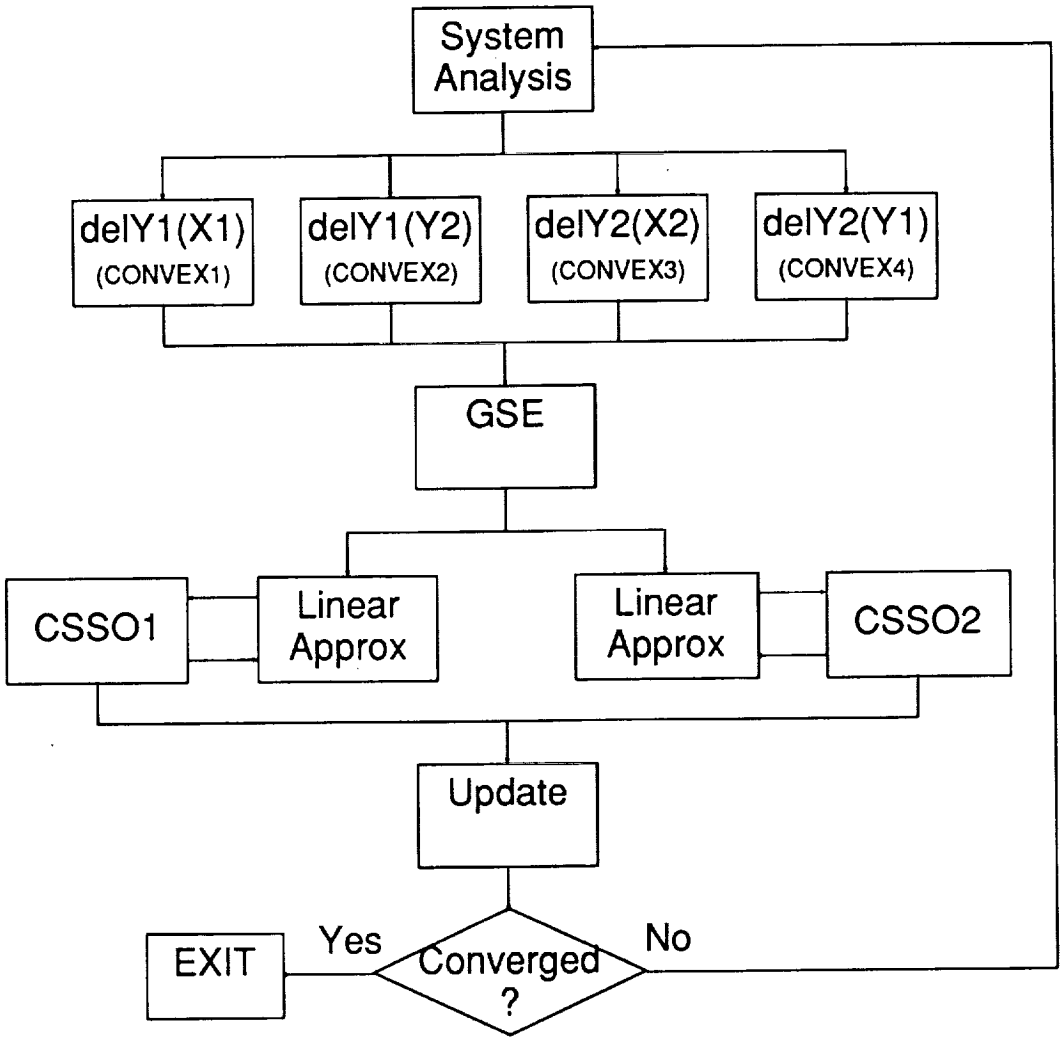


Figure 7.1 Distributed processing flowchart in CSSO.

violations. Since the t coefficients essentially allow for a possible violation of constraints in one subspace as long as an oversatisfaction of that constraint is obtained in the other, bounding the t coefficients has the effect of limiting the violation that can potentially occur. The effect of forcing the r coefficients to be active for the last few optimization cycles is also investigated. This application forces each subspace optimization to satisfy all constraints rather than allowing for a trade-off to occur between subspaces.

Variable Move Limit Strategy

Designers generally adopt a move limit strategy which involves initially assigning all design variables the same move limit value. As the design process progresses and the optimum is approached, the initial move limit value is continually reduced as the approximations become more critical for convergence. However, it may not be reasonable to group all design variables together in assigning move limit values. If certain design variables can be identified as having the most impact on a design and therefore requiring more restrictive move limits, it would be possible to allow the less critical design variables more leeway in their associated move limits. This would result in potentially reducing the overall cycles required for convergence, thus increasing computational efficiency. In this application, implementation of a variable move limit strategy is made to achieve such an efficient convergence scheme.

The effectiveness of each design variable is quantified by means of the previously defined effectiveness coefficients with the difference that the K.S. function cited in this case is a composite K.S. function representative of the most critical cumulative constraints from each subspace. The resulting effectiveness coefficients define an effectiveness space, in which upper and lower bounds must be determined as follows.

The mean value of the effectiveness space can be determined from

$$\bar{e} = \frac{1}{N} \sum_{i=1}^{N} e_i \tag{7.22}$$

where N is the total number of design variables. The associated standard deviation is determined from the relation,

$$\sigma(e) = \left[\frac{1}{N-1} \sum_{i=1}^{N} (e_i - \bar{e})^2 \right]^{1/2} \tag{7.23}$$

The upper and lower bounds of the effectiveness space can now be defined in terms of the mean value and standard deviation as

$$\begin{aligned} e^u &= \bar{e} + \sigma(e) \\ e^l &= \bar{e} - \sigma(e) \end{aligned} \tag{7.24}$$

Once the effectiveness space bounds are defined, associated move limits can be assigned to each design variable. For instance, the upper and lower effectiveness bounds might correspond to areas beyond which 90% and 10% move limits are permitted, respectively. Design variables with effectiveness coefficients falling within the upper and lower bounds would be assigned move limits based on a linear distribution between the bounds.

A description of a verification procedure to demonstrate the feasibility of the variable move limit strategy is presented in Appendix A. Examples and a discussion of results obtained for various applications are discussed.

Concurrent Subspace Optimization - Embedded Expert System Method

Distributed Processing Environment

A parallel computing environment is utilized in which the structural analysis is performed on a CONVEX computer while the flight mechanics analysis is carried out on a

VAX/VMS system. The coordinating node which is responsible for delegating tasks between these environments is also a VAX/VMS system.

Design Variable Allocation

Due to the interactions which exist in a highly coupled problem, such as in multidisciplinary applications, it is reasonable to assume that certain design variables may contribute to more than one subsystem analysis. Although these design variables can be allocated to subsystems based solely on sensitivity information, heuristics pertaining to traditional allocations are useful in situations for which no dominant choice surfaces.

In the CSSO-EES method, the design variable allocation proceeds as follows. Based on the sensitivity analysis, the impact of each design variable on a particular subsystem's outputs is quantified using effectiveness coefficients, $e_{i.}$. Based on the values of these effectiveness coefficients, the relative importance of each design variable with respect to each subsystem's analysis is established by a rank-ordering procedure. Design variables which demonstrate a relatively similar influence on more than one subsystem are subjected to evaluation by the embedded expert system capability to determine their subsequent allocation.

The facts which must be asserted into the fact-list include the total number of design variables, the number of design variables already allocated to the various subsystems, and the type of design variable that is being considered for allocation. For example, in the structures/controls problem, the design variables are either sizing or damping variables. If the variable under question is a sizing variable, the fact would be entered into the fact-list with the command

```
(assert      (dv   sizing))
```

The rules in the knowledge base follow the logic that if the number of design variables already allocated to a particular subsystem is large, then the questionable design variable should be allocated to the non-traditional subsystem. For instance, if the number of design variables already allocated to the structures subsystem exceeds 60% of the total number of design variables, then even though the variable to be allocated is a sizing variable, it will be allocated to the controls subsystem in order to more evenly distribute the variables before performing the subspace optimizations. Figure 7.2 demonstrates the logic tree for the design variable (sizing variables) allocation knowledge base.

Optimization Parameter Determination

The program CONMIN [Van73], a constrained minimization code based on the usable-feasible search direction algorithm, is used to perform the subspace optimizations. Certain parameters associated with the algorithm must be prescribed by the designer prior to implementation. The expertise required to assign meaningful values to these parameters is often a limiting factor in the method's use. The implementation of an expert systems capability that monitors the progress of the algorithm and dynamically adjusts the parameters under question permits use of the algorithm by general designers. The parameters that are used in this implementation are 'itmax', 'delfun', 'dabfun', 'itrm', 'phi', 'theta', and 'ct'.

The 'itmax' parameter corresponds to the maximum number of iterations in the piecewise linear optimization process. 'Delfun' is the minimum relative change in the objective to indicate convergence, whereas, 'dabfun' is the absolute change in the objective function. The 'itrm' parameter controls the number of consecutive iterations required for convergence. 'Phi' is a participation coefficient that is used for infeasible designs and corresponds to how hard a design will be 'pushed' towards the feasible region. 'Theta' is

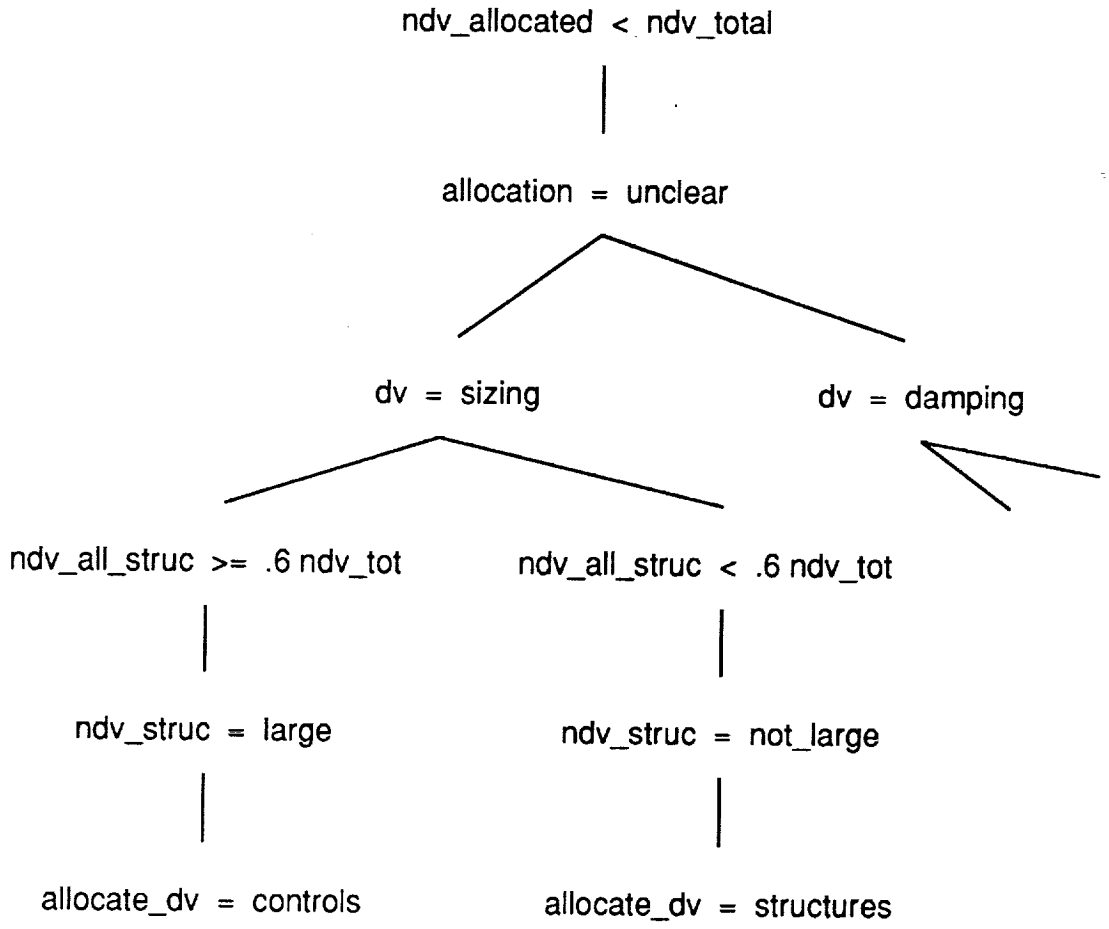


Figure 7.2 Logic tree for design variable allocation.

essentially the mean value of the push-off factor. 'Ct' is a constraint thickness parameter used to define whether constraints are active or inactive.

The embedded expert system capability in the subspace optimizations addresses the situation in which the optimization termination is governed by the fact that the number of iterations has reached the maximum number prescribed by the user ('itmax'), rather than that convergence is achieved. If itmax is reached and the solution corresponding to the final iteration is feasible, then the value of 'itmax' is doubled, unless this new value exceeds an upper bound, in which case the convergence criteria is investigated. If the parameters 'delfun' and 'dabfun' are both less than prescribed upper bounds, then these values are increased. If either of these bounds are exceeded, the parameter 'itrm' is reduced as long as it has not reached a lower bound. If this parameter has reached the prescribed lower bound, the fact 'process stop' is asserted into the fact-list. A failure to achieve a converged solution when the design is feasible after adjusting the above convergence parameters suggests a formulation or input problem.

If, on the other hand, the number of iterations is equal to 'itmax', but the solution corresponding to the final iteration is infeasible, then those parameters associated with the constraints are investigated. If 'phi' has not yet been increased in previous applications, then its value is tripled, thereby "pushing" the design towards the feasible region three times harder. If the value of 'phi' is greater then the initial value, but both 'phi' and the value of 'itmax' are less than prescribed upper bounds, then these parameters are increased in value. If 'phi' or 'itmax' exceed their upper limits, however, then the 'theta' and 'ct' parameters are modified based on the assumption that the infeasibility and nonconvergence problem is associated with highly nonlinear constraints. If both parameters are less than upper bounds, new values of these parameters are calculated. Otherwise, the fact 'process stop' is asserted into the fact-list, as a continued failure to achieve a converged, feasible solution once again suggests a user-associated problem, rather than one resulting from

algorithmic performance. Figure 7.3 shows the logic sequence for this heuristic implementation.

Variable Move Limit Strategy

The strategy proceeds as described in the previous section with the exception that heuristics are applied to refine upper move limit bound that is originally prescribed by the user. The facts which must be asserted into the fact-list include constraint satisfaction status from previous cycles, feasibility status of the converged solutions within the subspace optimizations, and bound adjustment information from previous cycles. The rules use these facts to determine whether adjustments to the prescribed move limit bounds are warranted. If constraints were violated after the update following the subspace optimizations (in previous cycles) in which converged feasible solutions were found, then bounds will be tightened. If the upper bound had already been adjusted on the side of conservativism in previous cycles, it must be adjusted even more strictly for the present cycle. If, on the other hand, constraints were consecutively satisfied following an update, the upper bound can be adjusted for leniency. Once the bounds are established, move limits for the design variables whose effectiveness coefficients fall within the effectiveness space are determined based on a linear distribution, from the equation

$$ml\% = \frac{\left(e - e^{l}\right)}{2\sigma(e)}\left(ml^{u} - ml^{l}\right) + ml^{l} \tag{7.25}$$

where ml^{u} and ml^{l} are the upper and lower move limit values originally prescribed by the user with the upper bound modified by the embedded expert system as the design progresses (i.e. 90% and 10%). Figure 7.4 shows the decision tree for the move limit strategy rules.

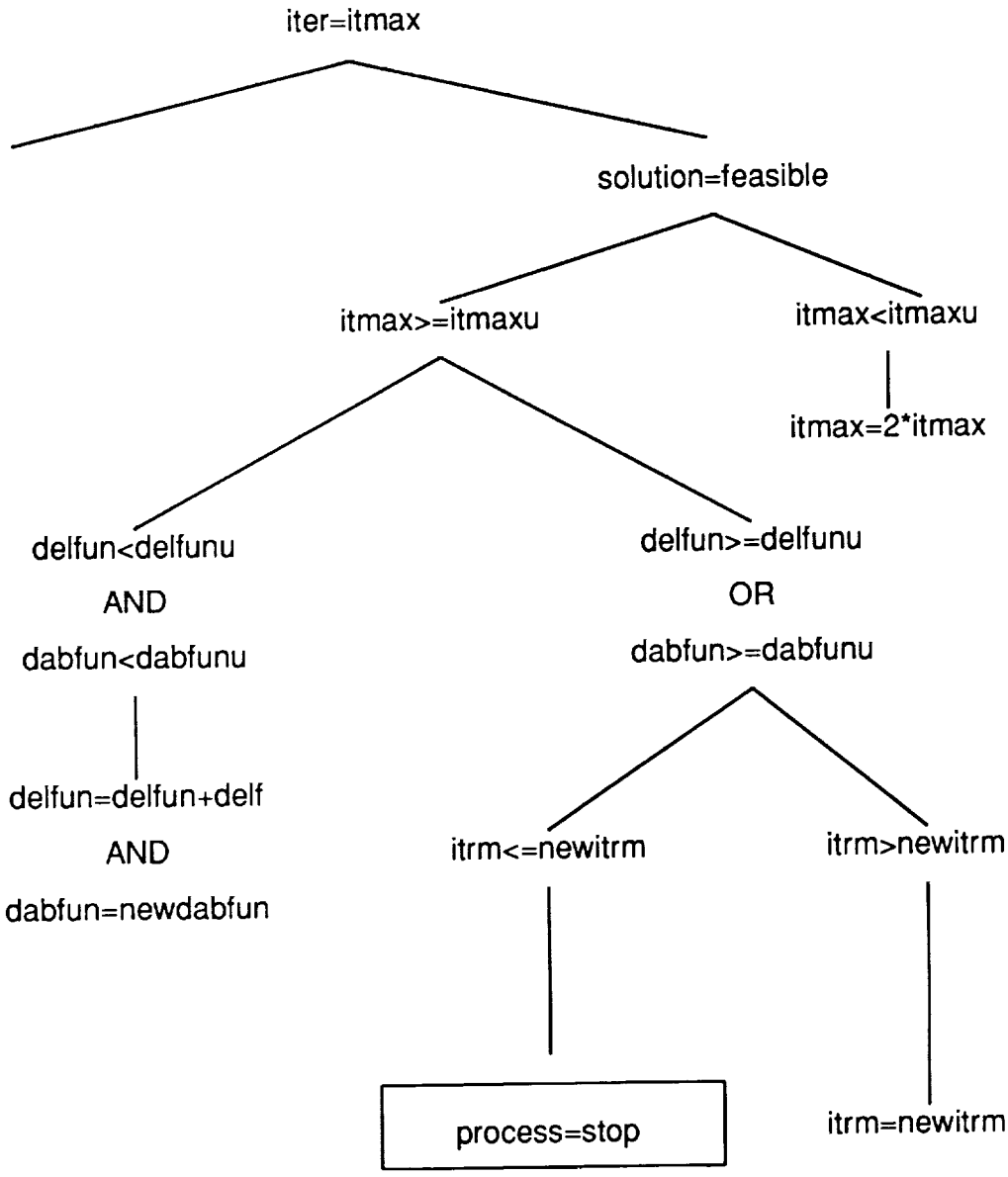


Figure 7.3 Logic tree for optimization parameter determination.

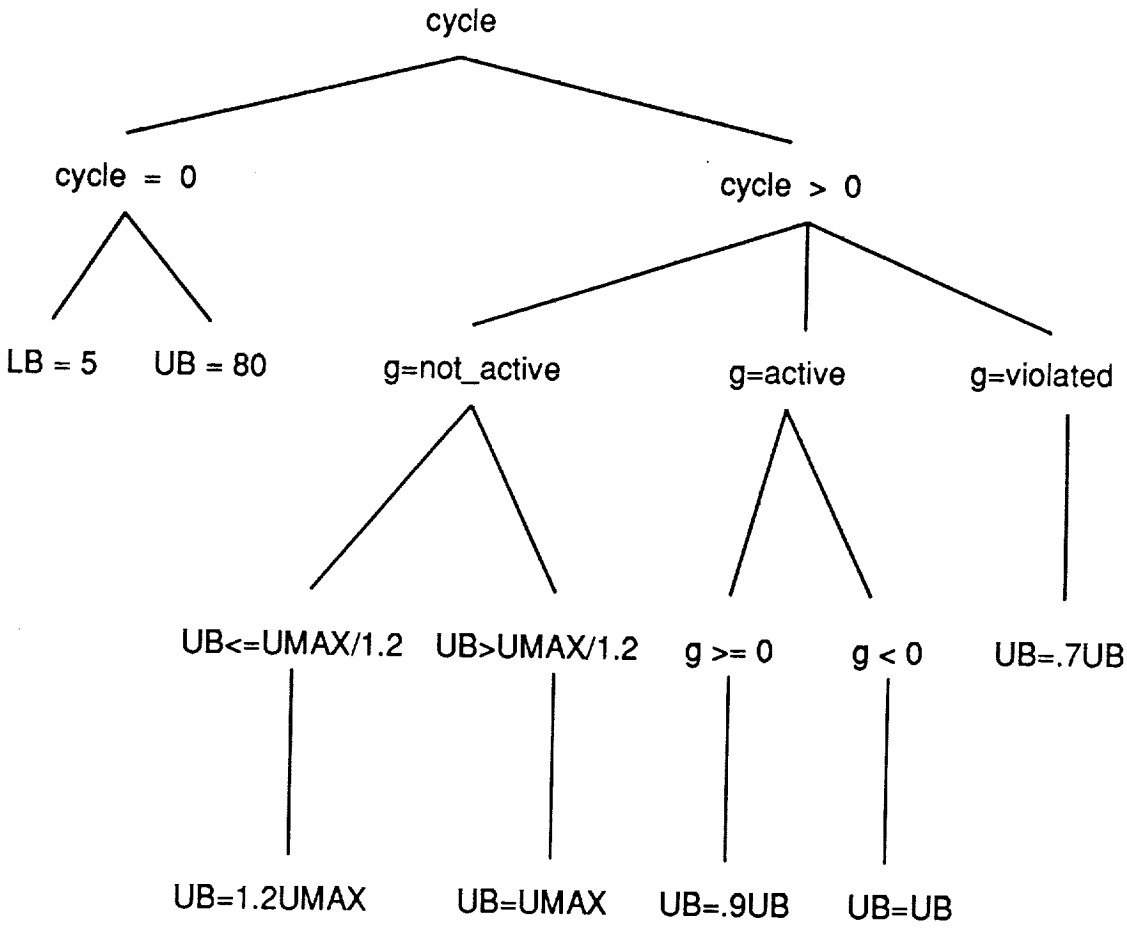


Figure 7.4 Logic tree for heuristic-based variable move limit strategy.

Coordination Coefficient Assignment

Coordination variable assignment based on heuristics as opposed to a COP alleviates problems associated with obtaining problem parameter sensitivities, such as premature convergence to suboptimal designs or active constraint switching. The knowledge base contains rules to determine whether a trade-off will be permitted in the present cycle, based on constraint satisfaction histories from the present and previous cycles. If no trade-off is permitted, the r coefficients are determined so as to assign a greater responsibility for a cumulative constraint satisfaction to those SSOs that have a relatively greater influence on that constraint. Values for these coefficients are algebraically determined from a scaling process involving manipulation of sensitivity information obtained from the GSE. If a trade-off is permitted, then the subspace in which the violation is to be permitted is determined based upon each subspaces ability to reduce the system objective function. The subspace with the ability to reduce the objective function by the greatest amount is permitted either a 20% or 10% constraint violation depending on whether the constraint value of the present cycle is more satisfied than the previous cycle or not.

The facts which must be asserted into the fact-list include information pertaining to traditional constraint allocations and the numbers of design variables allocated to the various subspaces. The rules use these facts to determine whether modifications to the basis should be implemented. A major consideration involves the traditional allocations of constraints. For instance, if the constraint under question limits the allowable stresses in the truss members, it would normally be associated with the structures discipline. The responsibility for satisfying it would traditionally rest with the sizing variables within that discipline. However, if the number of variables allocated to the traditional subspace is for some reason unusually small, difficulties would possibly arise with satisfying the

constraint within that subspace. Therefore, the responsibility for satisfying that constraint is shifted to the non-traditional subspaces.

Figure 7.5 demonstrates the logic for determining switch parameters and trade-off coefficient values for subspace 1. Appendix B contains the rules for the four heuristic implementations.

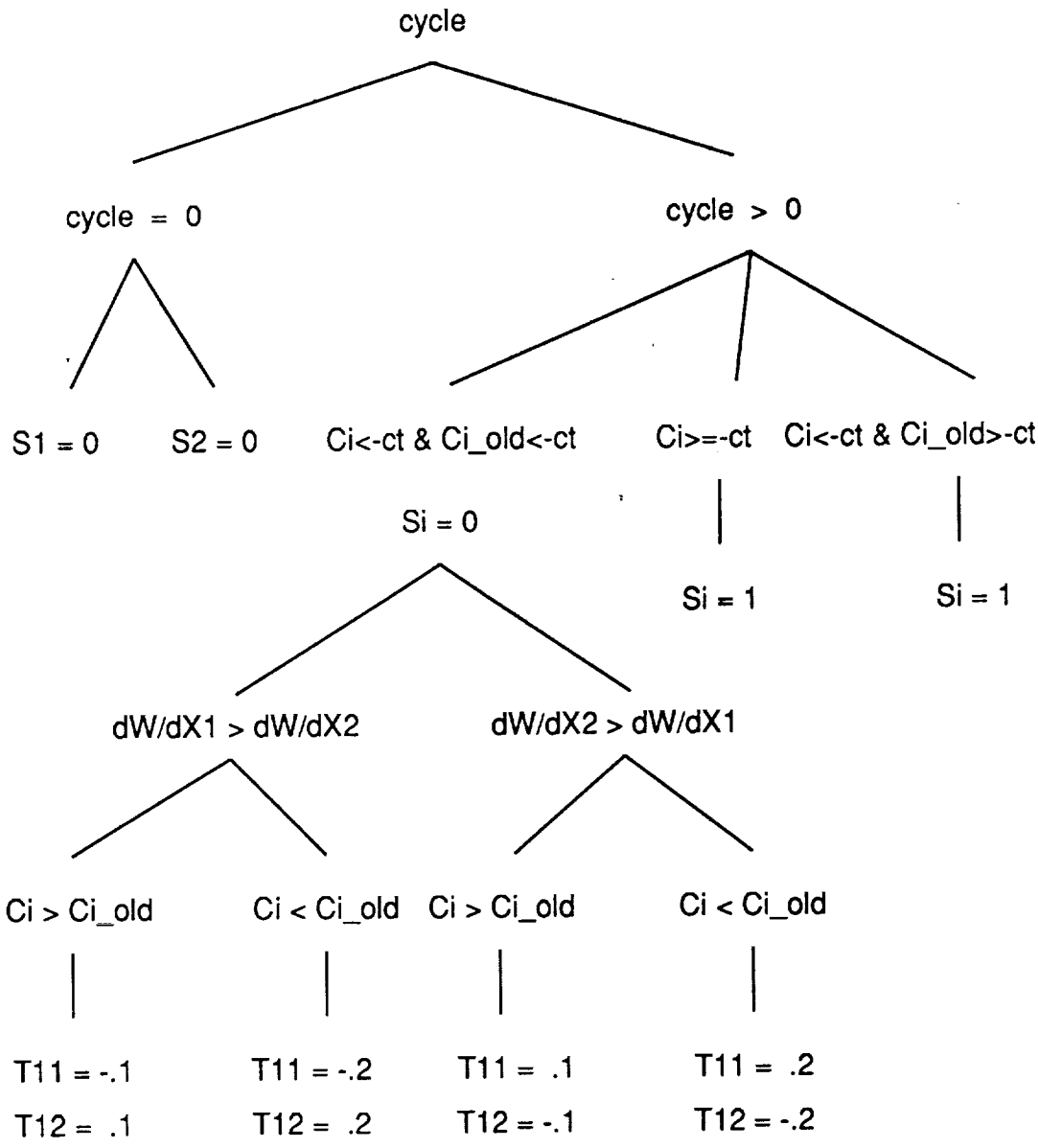


Figure 7.5 Logic tree for heuristic coordination coefficient determination.

C-2

CHAPTER 8
DISCUSSION OF RESULTS

Results obtained by implementation of the solution and verification techniques described in Chapter 7 are discussed for each decomposition method.

Global Sensitivity Equation Method

The effect of constraint representation, design variable allocation, and normalization of the GSM on condition number can be seen in Figures 8.1a and 8.1b. As expected, the condition number increased with increased dimensionality of the GSM and was unacceptably large when normalization was not used. The allocation of design variables to distinct disciplines and their inclusion in the output vector had little influence on the condition number. As can be seen in these figures, implementation of the normalization scheme described previously effectively reduced condition number, thus contributing to improved solution accuracy.

The effect of normalization of the GSM, constraint representation, dimensionality, and design variable allocation on computational requirements is summarized in Figures 8.2a-8.2c. Although Figures 8.2a and 8.2b demonstrate little change in solution time for direct decomposition with normalization, Figure 8.2c shows significant reductions when used with Gauss-Siedel iteration. For the limited dimensionality problems considered in this work, these solution times were comparable to those required in a direct decomposition approach. Due to roundoff error accumulation in a direct decomposition approach, the iterative strategy is preferred. In the iterative framework of the design synthesis process, it is possible to use the solution of the previous iteration as the initial choice for the current

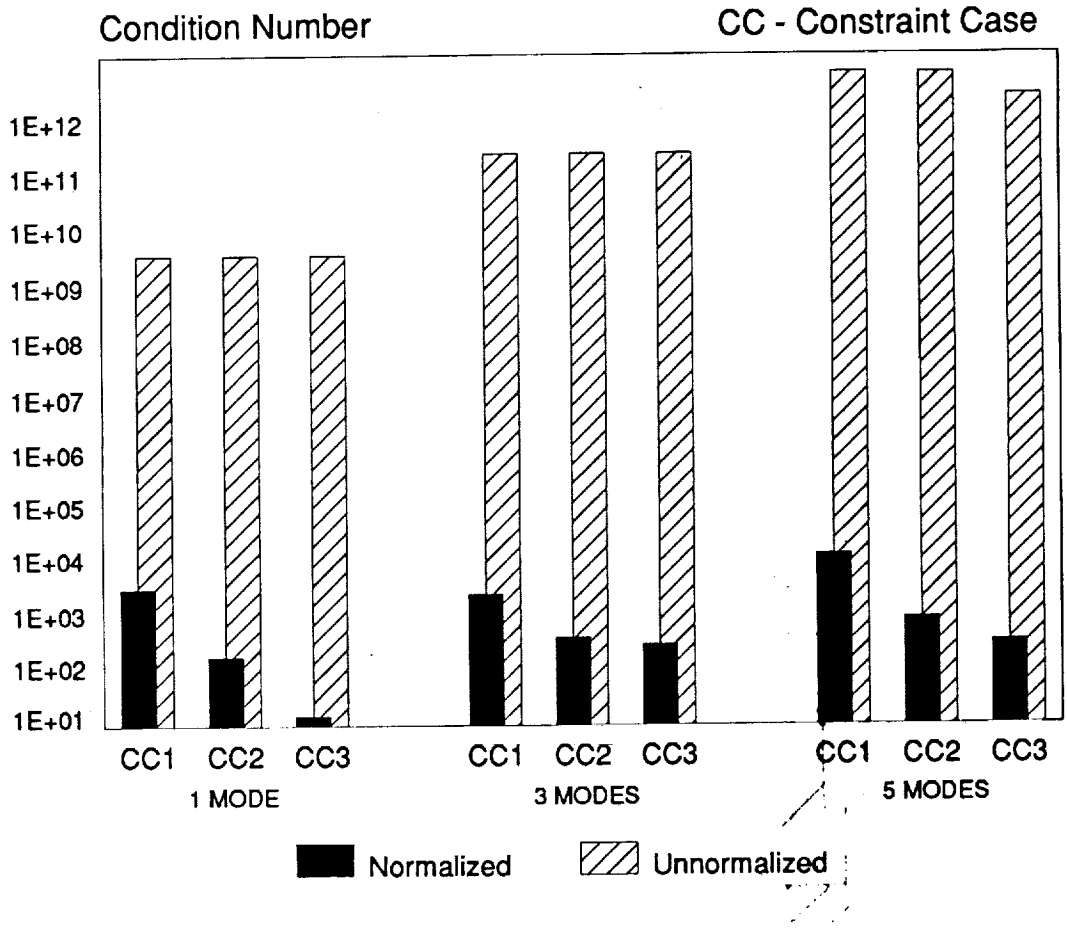


Figure 8.1a Condition number variation with constraint representation, normalization, and dimensionality.

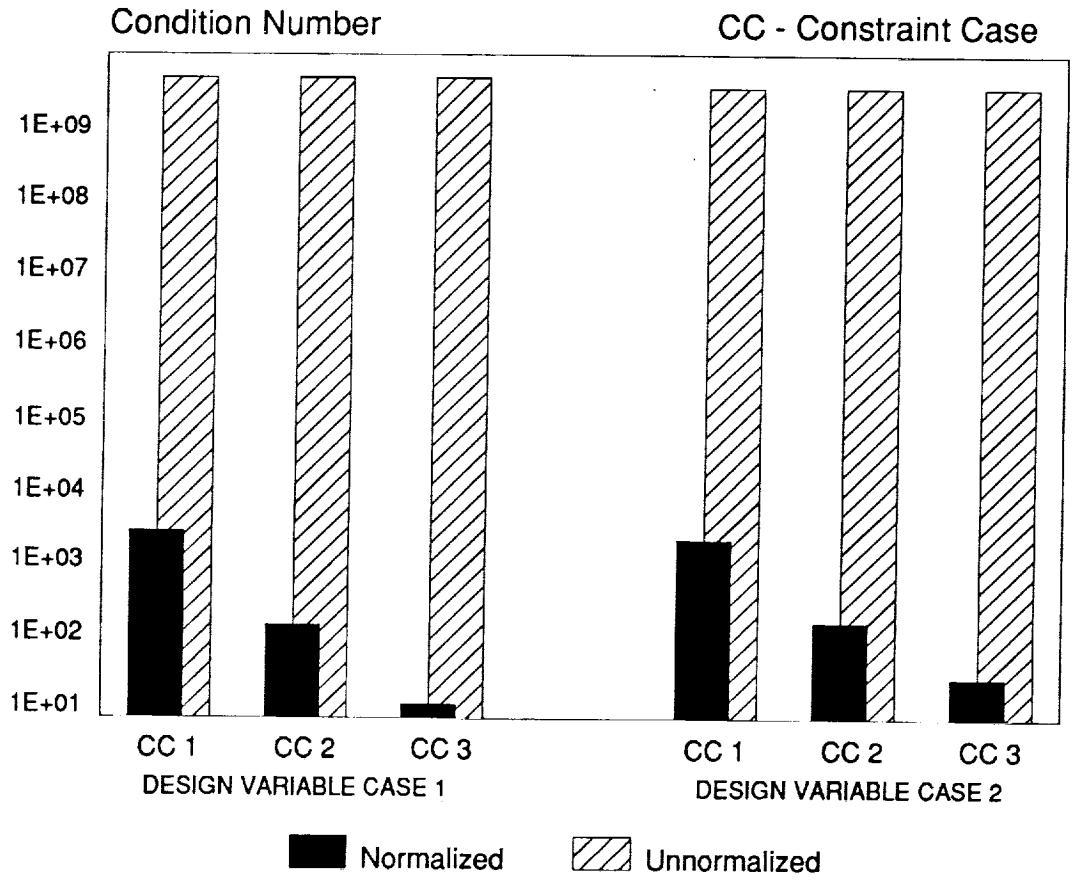


Figure 8.1b Condition number variation with constraint and design variable representation and normalization.

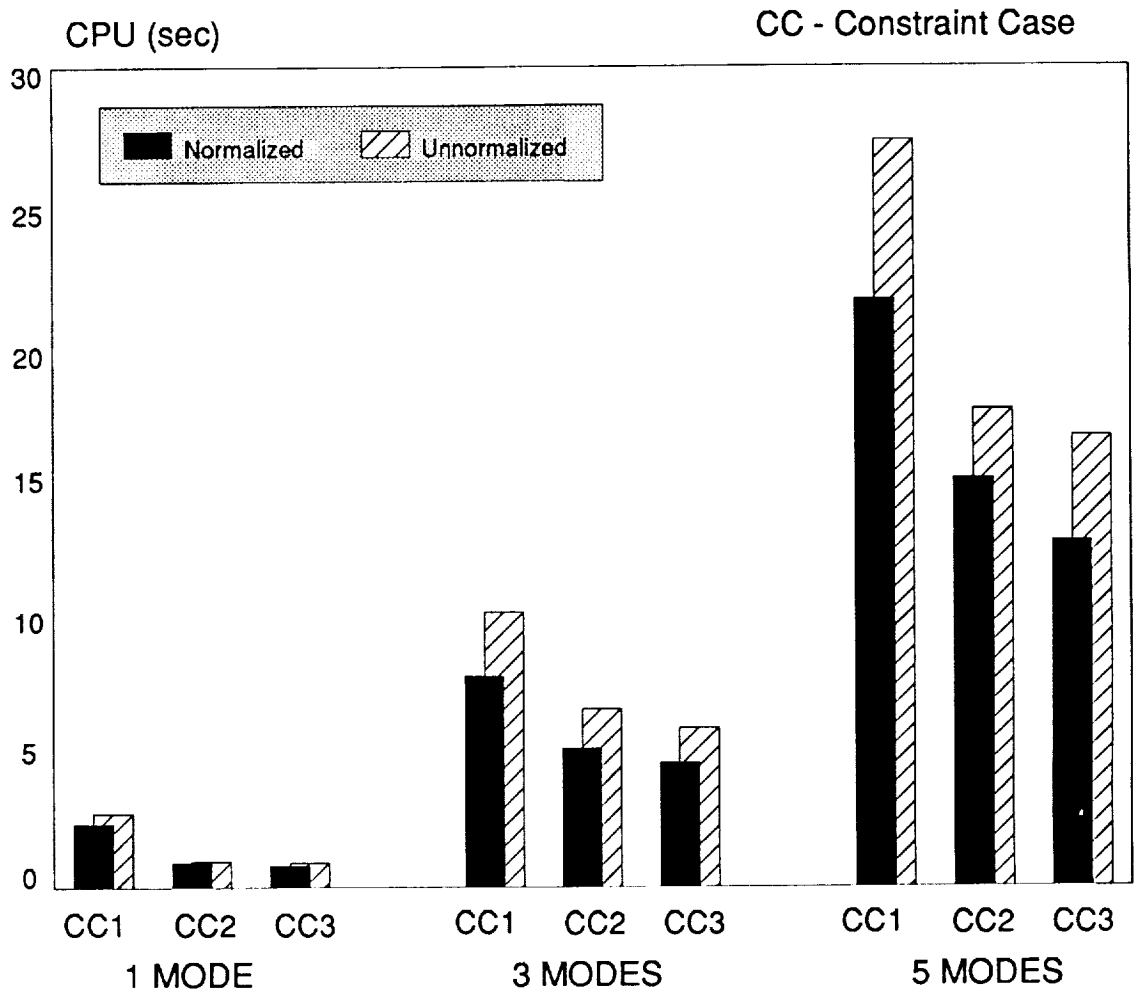


Figure 8.2a Computational time variation with constraint representation, normalization, and dimensionality for direct decomposition solutions.

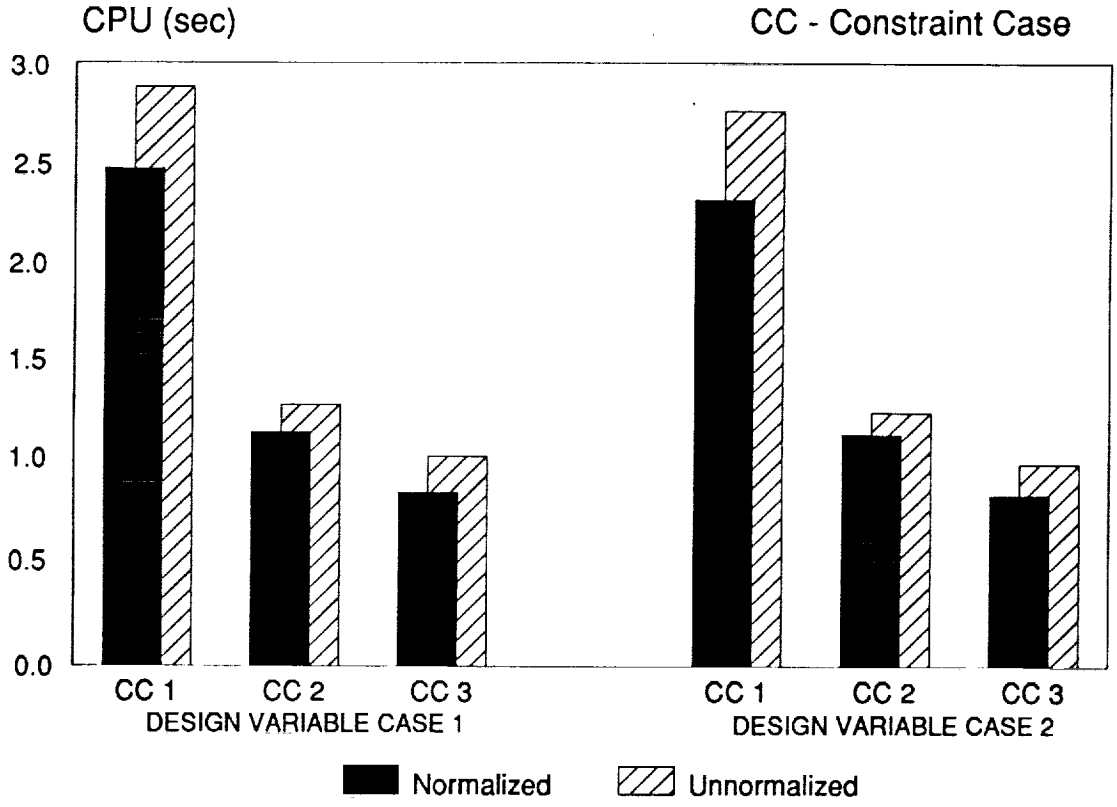


Figure 8.2b Computational time variation with constraint and design variable representation, and normalization for direct decomposition solution.

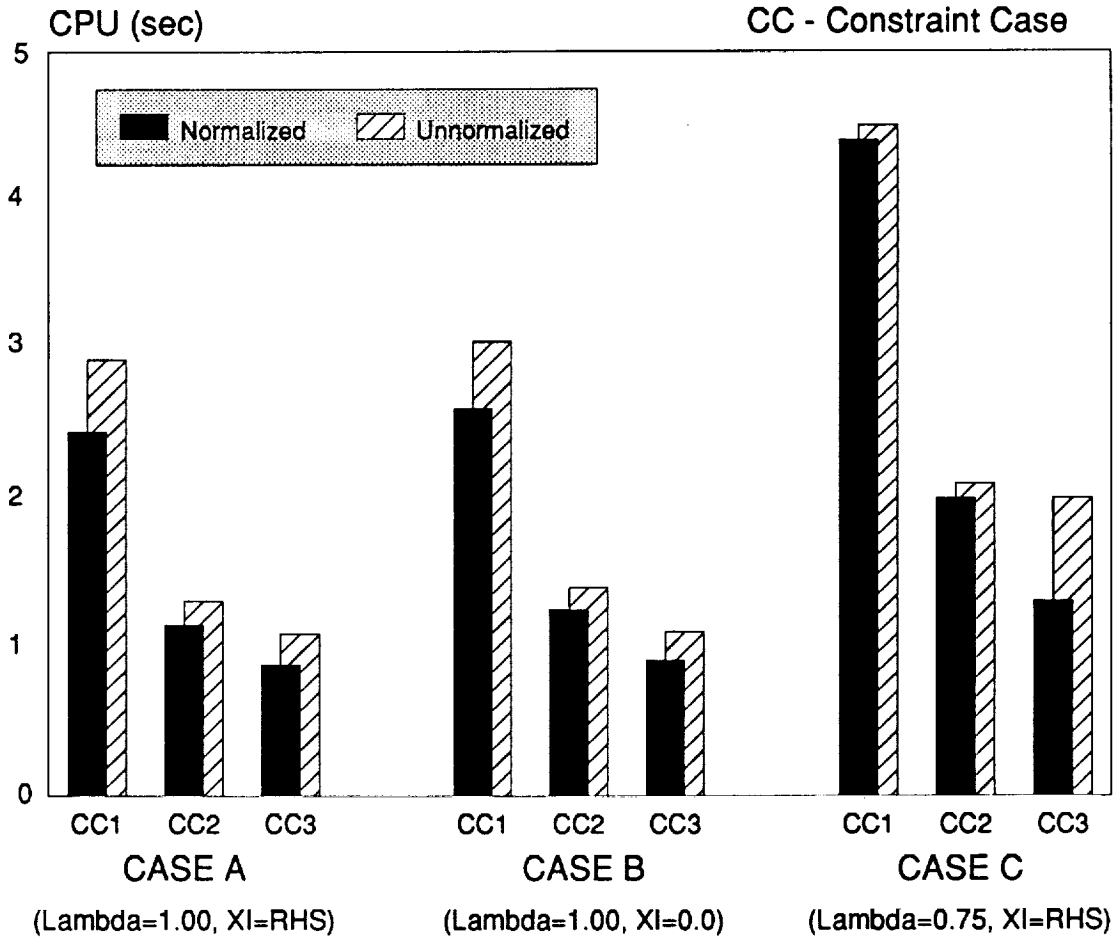


Figure 8.2c Computational time variation with constraint representation, normalization, and dimensionality for iterative solution.

iteration. Mixed results were obtained upon application of such a strategy, failing to conclusively establish any distinct advantage in computational savings.

Results for the output response sensitivities were obtained by direct decomposition and iterative solution approaches, as well as by a finite difference technique using the coupled system equations. A comparison of these results in terms of the standard deviation measure previously defined for various constraint and design variable cases are summarized in Figures 8.3a-8.3e. Extremely good agreement between iterative and direct decomposition solutions is shown with slightly improved agreement with the use of normalization implementation. A comparison between direct decomposition and finite difference solutions shows good agreement with increased deviations resulting from the somewhat loose convergence criteria used in the solution of the coupled analysis equations in the finite difference approach. Direct decomposition solutions for the two design variable representation cases (Figure 8.3e) demonstrated reasonably good agreement between the two representations, but did not conclusively establish the advantages of one type of design variable representation over the other.

The sensitivity information obtained in the above analysis was also used in a representative optimization application. Table 8.1 summarizes the initial and final designs for this exercise. Results obtained for the 1 mode, finite difference solution and the 3 mode, GSE solution are presented. The number of design variables for the 1 mode solution are less than the 3 mode solution due to the fewer gain components associated with the former set. The results, obtained at the end of the sixth iteration, demonstrated similar values for the design objective.

Concurrent Subspace Optimization Method

Since the CSSO is a newly proposed method, it is critical that optimization solutions obtained by application of this approach are compared to more established method

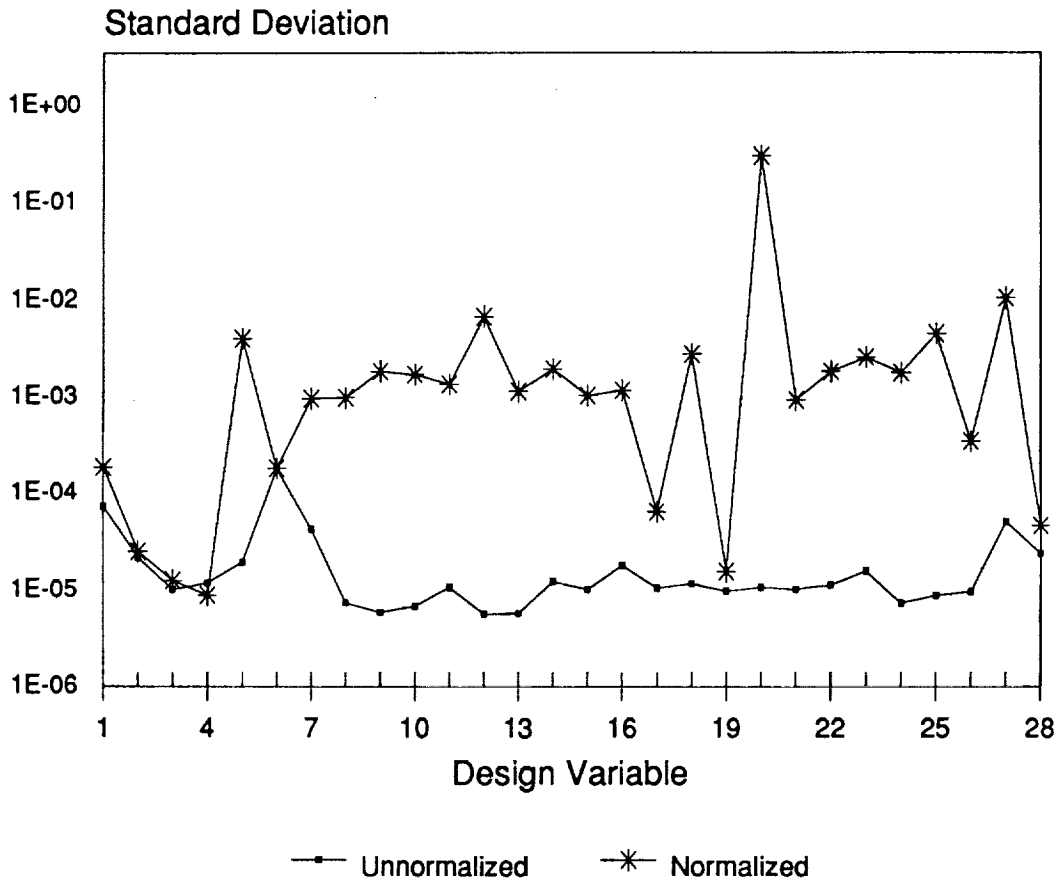


Figure 8.3a Effect of normalization on standard deviations between finite difference and direct decomposition solutions for design variable case 1.

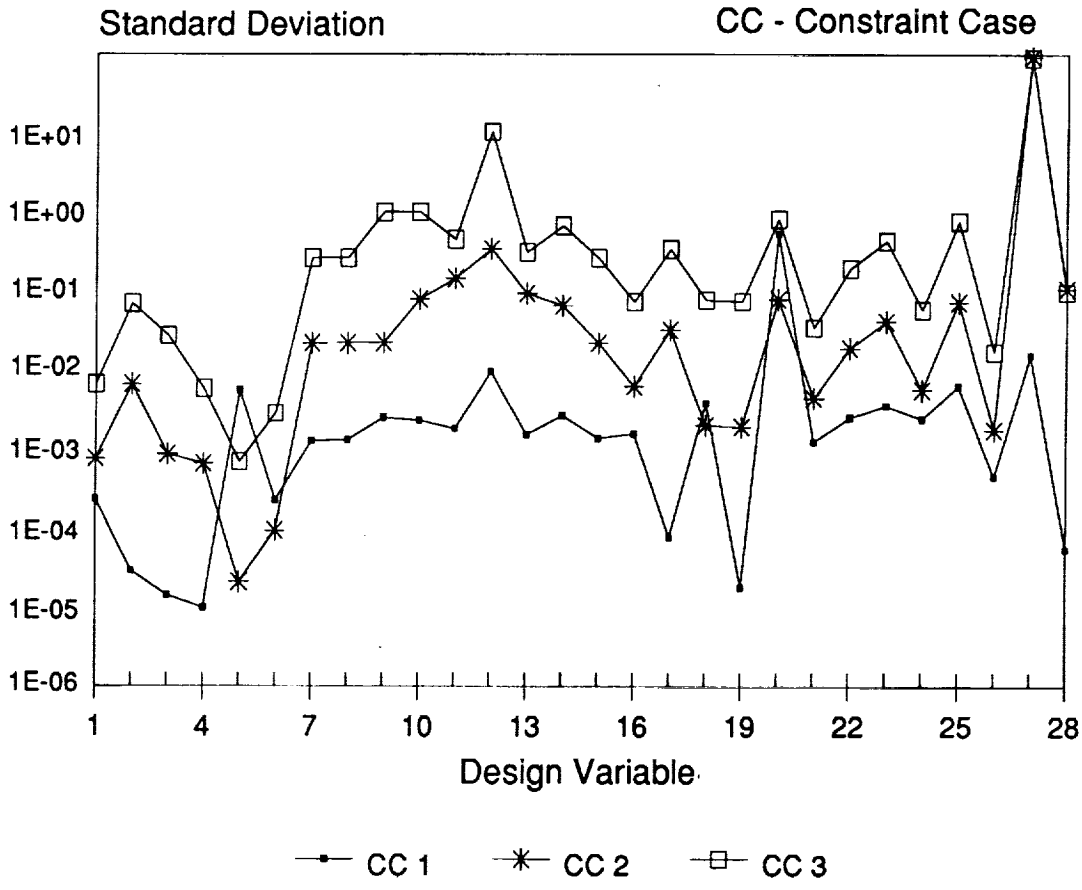


Figure 8.3b Effect of constraint representation on standard deviations between finite difference and direct decomposition solutions for design variable case 2.

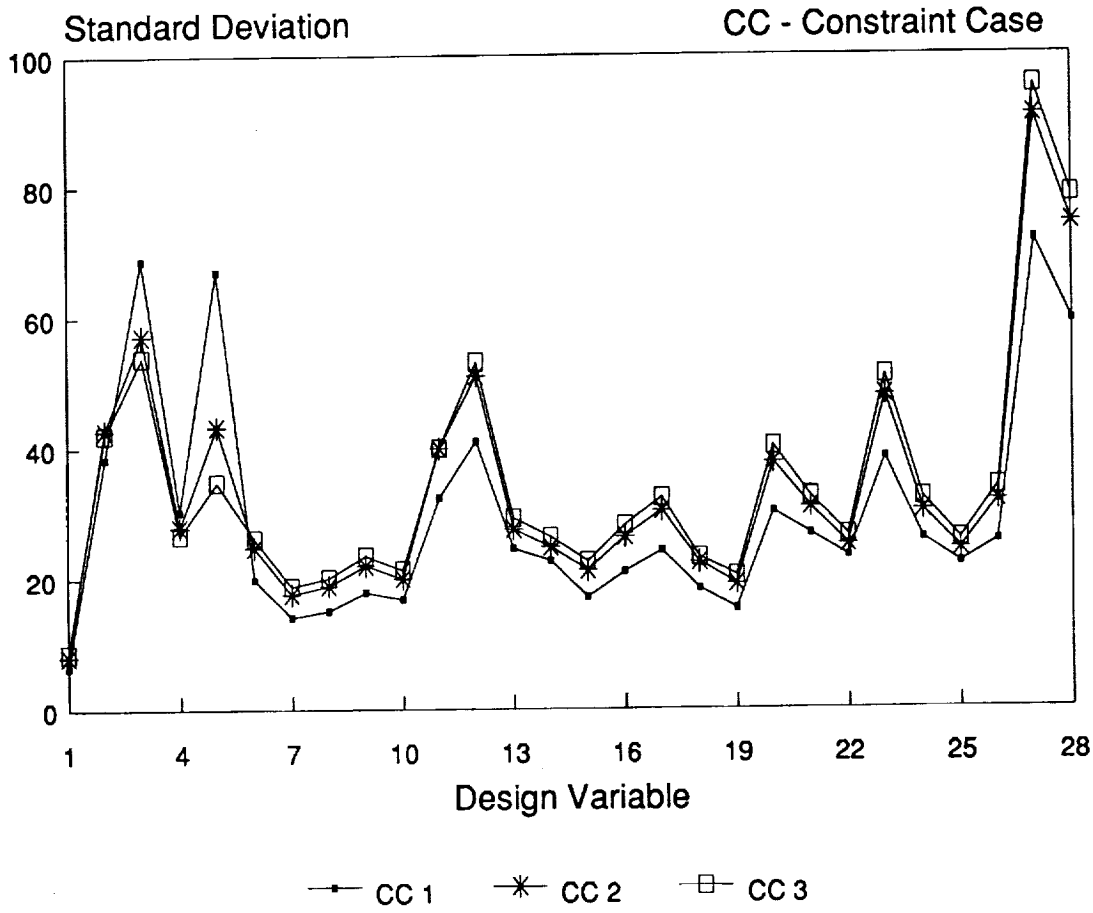


Figure 8.3c Effect of constraint representation on standard deviations between iterative and direct decomposition solutions.

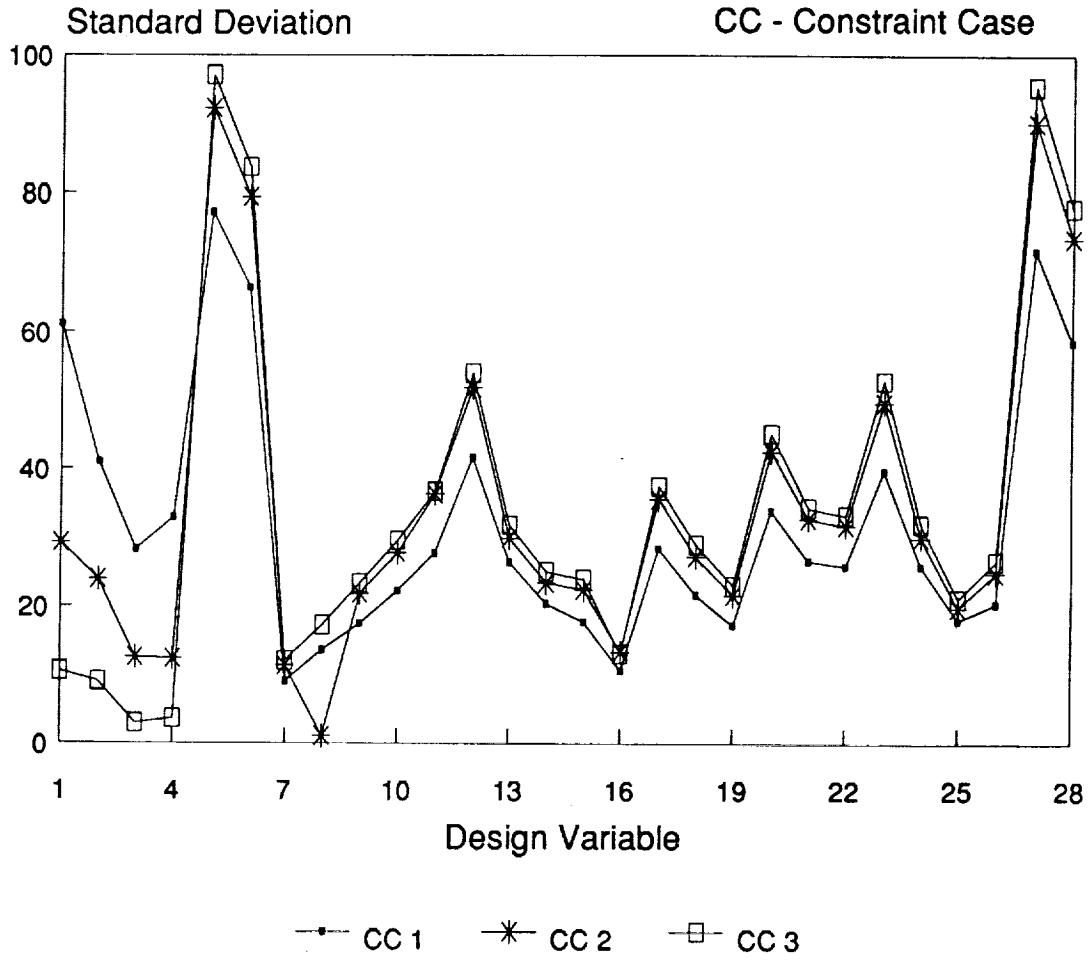


Figure 8.3d Effect of normalization on standard deviations between iterative and direct decomposition solutions.

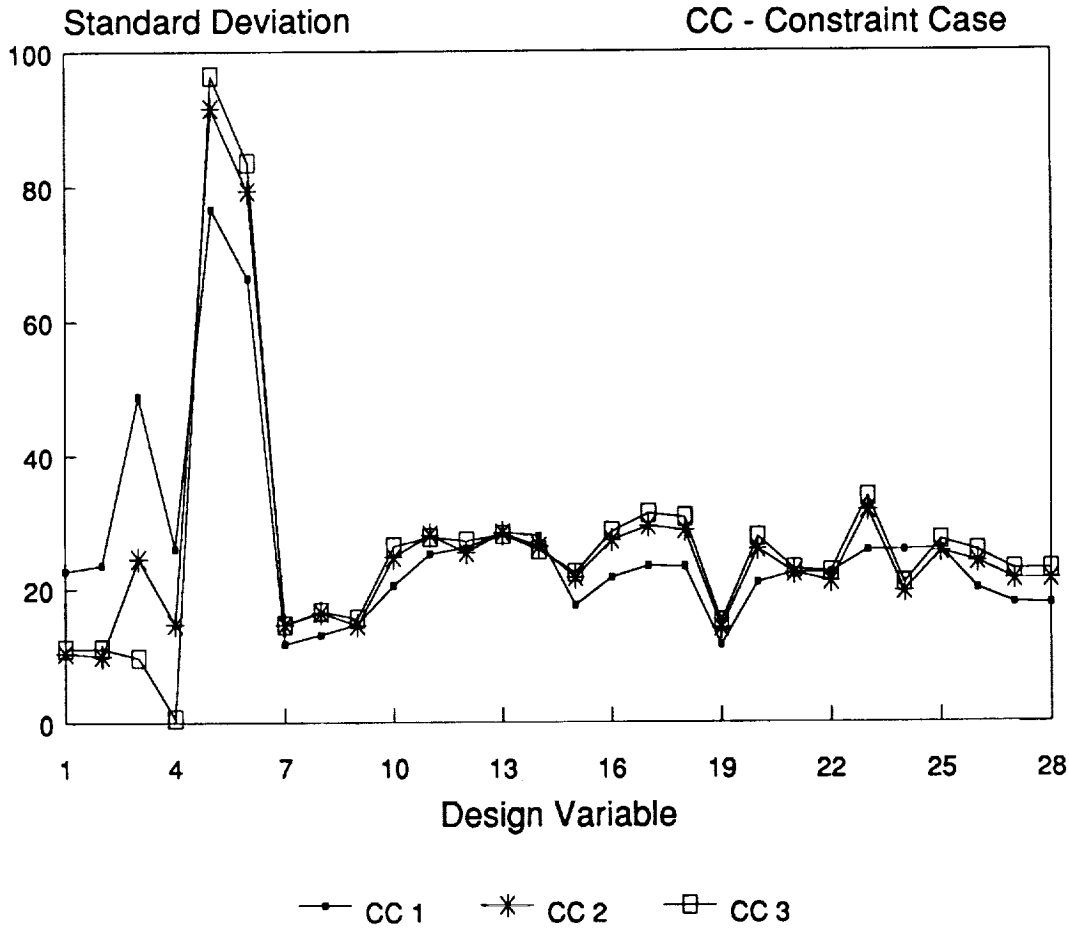


Figure 8.3e Effect of constraint representation on standard deviations between design variable representations for direct decomposition solutions.

Table 8.1 Summary of optimization results for GSE application to the aircraft synthesis problem.

DV	INITIAL	FINAL 1 mode, FD solution	FINAL 3 mode, GSE solution
1	64.00	77.1300	71.5629
2	64.00	73.9076	68.3836
3	44.50	45.5795	62.9380
4	217.0	166.129	133.937
5	1.500	0.57240	1.75480
6	123.5	108.545	108.175
7	0.400	0.23034	0.27464
8	0.400	0.17364	0.27381
9	0.300	0.15936	0.20981
10	0.300	0.14051	0.21033
11	0.200	0.15875	0.15450
12	0.200	0.11927	0.15462
13	0.200	0.15737	0.18509
14	0.200	0.15603	0.18503
15	0.050	0.02458	0.03286
16	0.050	0.02790	0.03262
17	0.050	0.03227	0.03745
18	0.050	0.01741	0.03198
19	0.050	0.01758	0.03205
20	0.050	0.02417	0.05509
21	0.040	0.01966	0.02981
22	0.040	0.02334	0.02988
23	0.030	0.03260	0.03104
24	0.030	0.02212	0.02902
25	0.020	0.02447	0.02900
26	0.020	0.02881	0.02513
27	0.100	0.11445	0.10003
28	0.100	0.11949	0.09965
29	0.100		0.10000
30	0.100		0.10132
31	0.100		0.07999
32	0.100		0.09999
OBJ	2033.37 lbs	1909.43 lbs	1888.58 lbs

solutions for verification purposes. To this end, various optimization solutions to the ten-bar truss problem described previously are presented in Table 8.2. The four sets of results presented correspond to optimization solutions where sensitivities were obtained by a finite difference procedure and by the GSE method (Cases 1 and 2, respectively), and by application of the CSSO method in both the traditional single processor and in a distributed processing environment (Cases 3 and 4, respectively). The CSSO solutions correspond to cases in which reciprocal approximations were used in conjunction with a variable move limit scheme. As can be seen from Table 8.2, all three methods (Cases 1-3) show good agreement, thus confirming the feasibility of the proposed CSSO method. Further, the ability to effectively parallelize the CSSO implementation was demonstrated by comparison of solutions corresponding to Case 4 with those obtained in Cases 1-3. Essentially the same design was found in all cases, with the distributed results demonstrating the versatility and potential computational efficiency of the CSSO method.

Figures 8.4a and 8.4b demonstrate that a reciprocal constraint approximation scheme results in largely improved constraint violation characteristics compared to a linear approximation scheme. The more accurate constraint approximations resulting from application of the reciprocal scheme allowed for larger move limits in the piecewise-linear approach at the subspace optimization level. Very little difference was obtained by implementation of the improved approximation scheme.

The outcome of a study to determine the effects of bounding the t coefficients is seen in Figures 8.5a and 8.5b, in which no upper bound and an upper bound of 1.0 was enforced, respectively. Twenty percent move limits were permitted and a reciprocal approximation scheme was used in both cases. It is obvious that the increase in the t coefficient value in the first case contributed to a continually worsening oscillatory convergence history. Applying an upper bound on the t coefficients resulted in a speedier convergence, as little time was spent compensating for constraint approximation inaccuracies occuring in prior cycles. Figure 8.5b also demonstrates the result of forcing

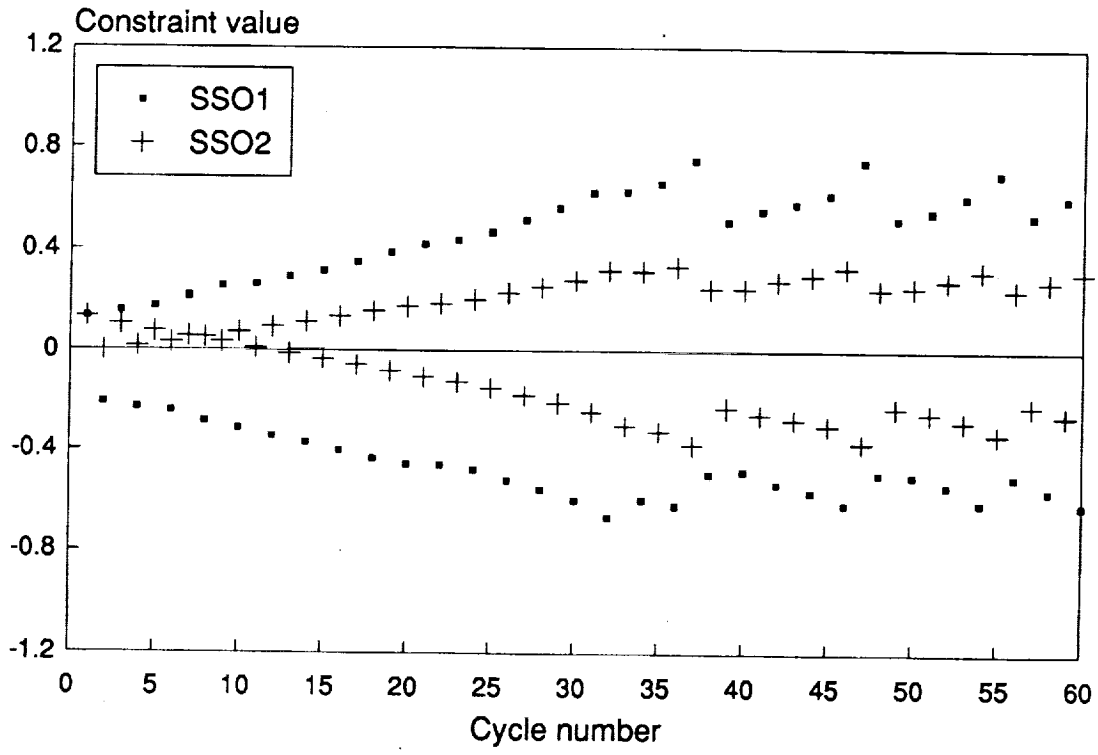


Figure 8.4a Constraint violation history for linear approximation scheme.

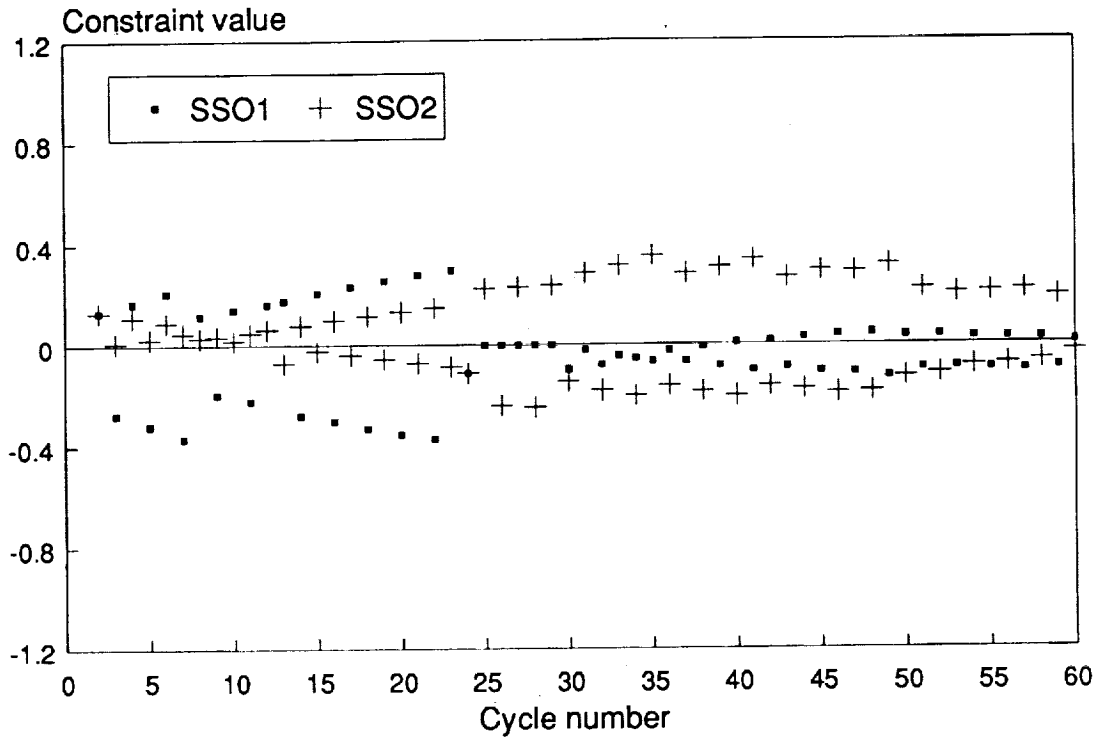


Figure 8.4b Constraint violation history for reciprocal approximation scheme.

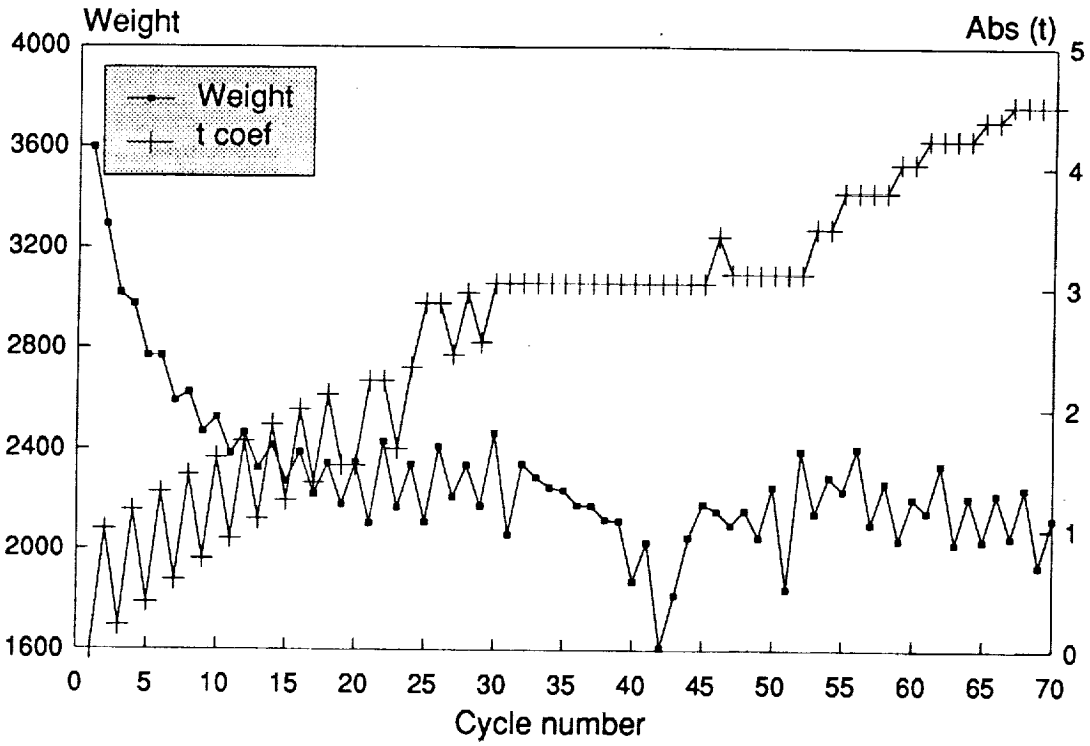


Figure 8.5a CSSO convergence history with unbounded t coefficients.

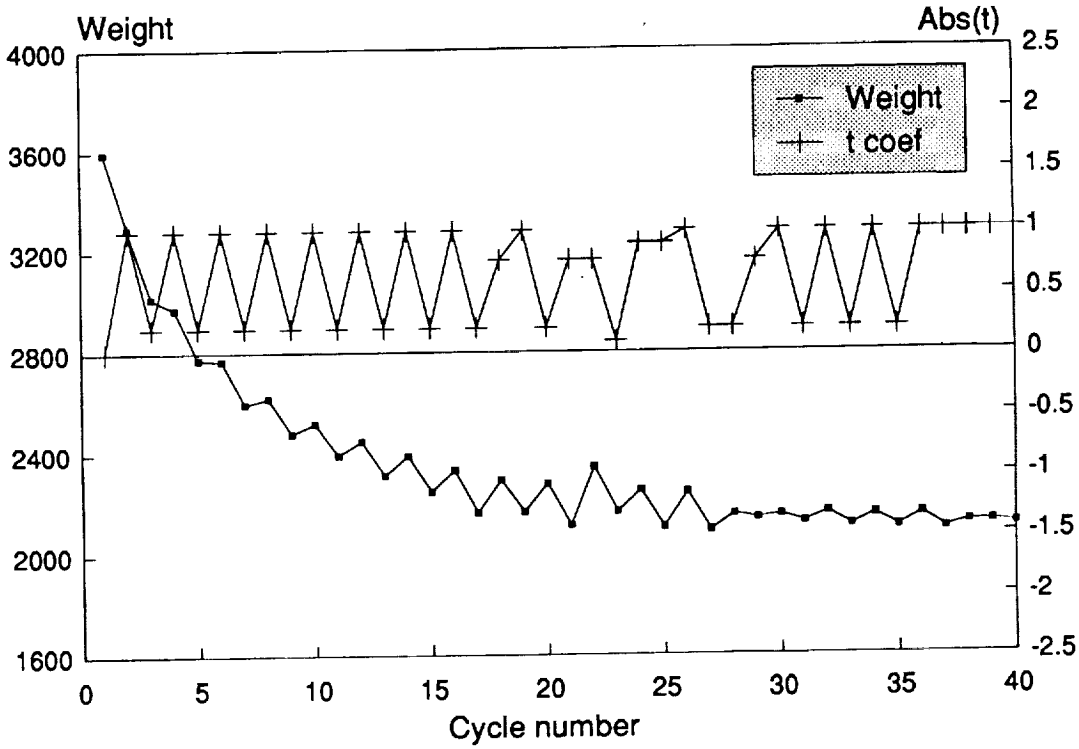


Figure 8.5b CSSO convergence history with bounded t coefficients.

Table 8.2 Comparison of initial and final optimization results for ten-bar truss model.

	INITIAL	FINAL			
		CASE1	CASE2	CASE3	CASE4
Weight (lb)	5601.3	2116.0	2115.6	2120.6	2113.1
DV (in^2)					
1	10.0	7.14	7.14	6.87	7.09
2	10.0	5.03	4.96	4.88	4.84
3	10.0	6.71	6.66	6.63	6.63
4	10.0	0.10	0.10	0.39	0.17
5	10.0	4.31	4.31	4.34	4.25
6	10.0	3.71	3.72	4.15	4.02
7	10.0	0.10	0.10	0.12	0.10
8	10.0	8.24	8.23	8.01	8.01
9	10.0	0.10	0.10	0.42	0.24
10	10.0	0.11	0.10	0.10	0.10
11	300.0	371.2	373.7	351.0	366.2

the r coefficients to be active in the final cycles of the design process. The trade-off capability of the t coefficients was 'turned off', resulting in a smooth convergence. Figure 8.6 shows the forced constraint satisfaction achieved by such an implementation.

Implementation of a variable move limit strategy based on the concept of effectiveness coefficients demonstrated substantially improved convergence characteristics. A comparison of the convergence history shown in Figure 8.5b with that resulting from an implementation of the heuristics-based variable move limit strategy in Figure 8.7, shows a marked improvement in objective function reduction after the first cycle. An overall reduction of twenty cycles was achieved by applying the move limit strategy, which translates into significant computational savings.

Concurrent Subspace Optimization - Embedded Expert System Method

Table 8.3 demonstrates results corresponding to four approaches for the control/structure interaction problem described in Chapter 6. Cases A and B correspond to 'all-in-one' optimization strategies in which the gradient information was obtained by the Finite Difference (FD) and Global Sensitivity Equation (GSE) approaches, respectively. Case C corresponds to the Concurrent Subspace Optimization method and Case D to the CSSO with the embedded expert system capabilities discussed in the previous section. Although final results are quite similar, there are definite differences between the results corresponding to the FD, GSE, and CSSO approaches. This difference can be largely attributed to the fact that the CSSO method makes extensive use of cumulative constraint representations. These functions have the effect of creating a constraint envelope which is slightly conservative, thus resulting in convergence to a slightly different final design. As can be seen from Table 8.3, only a 4% difference in the objective function exists between the CSSO and GSE method results.

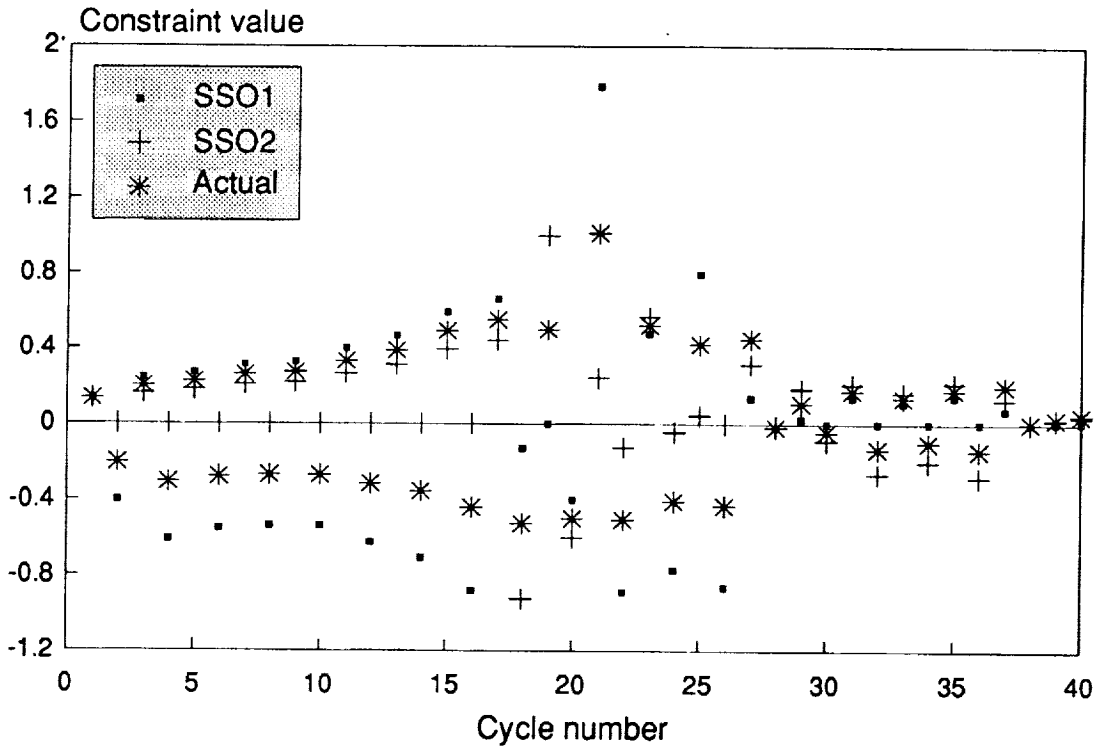


Figure 8.6 Constraint violation history with forced convergence.

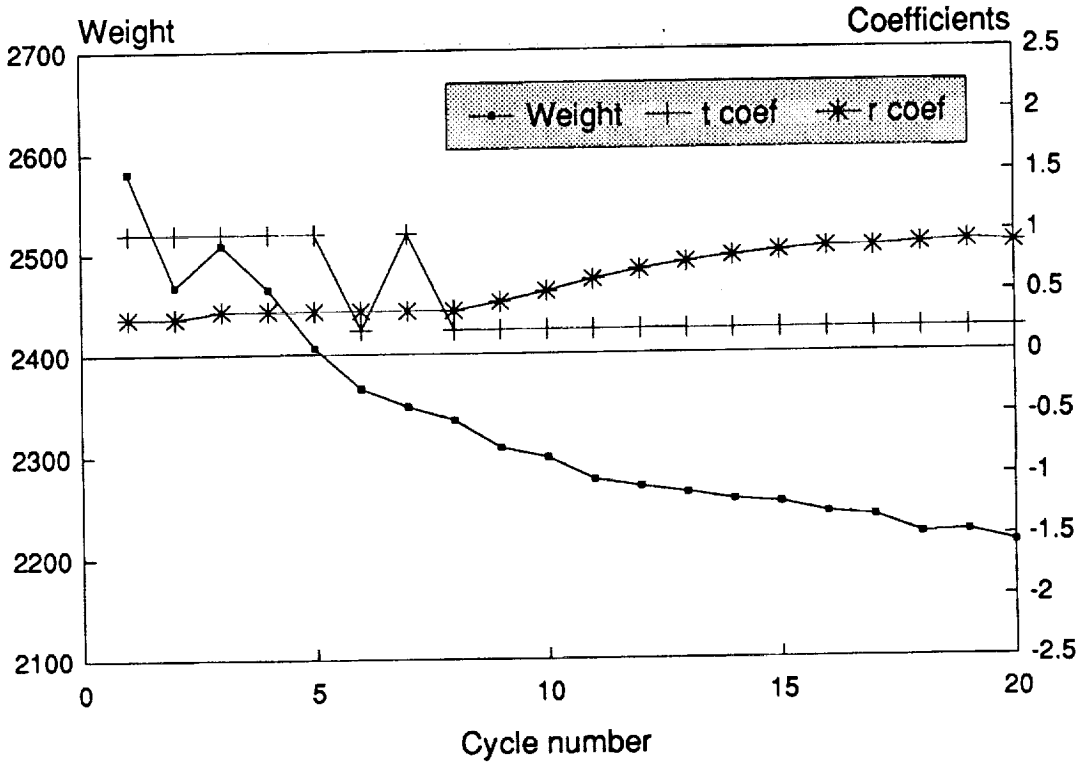


Figure 8.7 CSSO convergence history with variable move limit strategy.

Table 8.3 Comparison of optimization results for CSI problem.

	INITIAL	FINAL			
		FD	GSE	CSSO	CSSO-EES
Weight (lb)	848	410	427	447	451
DV (in^2)					
1	2.0	1.8	2.0	2.0	2.2
2	2.0	.91	1.0	.63	.95
3	2.0	2.0	2.1	2.2	2.3
4	2.0	.36	.39	.60	.78
5	2.0	.70	.82	1.3	.76
6	2.0	1.3	1.1	.83	1.1
7	2.0	.22	.20	.47	.30
8	2.0	1.4	1.3	.95	1.1
9	2.0	.54	.65	1.2	.59
10	2.0	.25	.38	.44	.56
11	.05	.06	.04	.02	.01
Cycle		12	15	29	25

The effect of reallocating the design variables after every five cycles was investigated for cases in which a strictly algorithmic allocation scheme (Case A) and the heuristics-based allocation scheme previously described (Case B) were used. Table 8.4 demonstrates the allocation of selected design variables (numbers 1, 5, 7, and 8) after 5, 10, and 15 cycles. The design variables were allocated to either subspace 1 (structures) or subspace 2 (controls). It is interesting to note that the weight after five cycles was 476 lb. for Case A and 460 lb. for Case B. Clearly, use of the heuristics-based design variable allocation scheme provided better convergence rates.

The embedded expert system capability was used effectively to determine values for parameters associated with the optimization code CONMIN. The capability both initialized parameters associated with linearity of the problem, as well as dynamically changed parameters for cases in which convergence was not achieved for the approximate optimization problem. The initialization process resulted in slightly increased values for 'phi', 'theta', and 'ct' since constraint non-linearities existed.

The incorporation of an efficient move limit strategy in which the upper bounds were heuristically varied resulted in a more efficient convergence scheme. As seen in Table 8.3, the CSSO method required 29 cycles to converge, whereas the CSSO-EES required 25. Figure 8.8 demonstrates the variation of the upper bound in the first 10 cycles, as well as the move limits associated with two representative design variables (number 4 and 6). As shown in the figure, the upper bound decreases as the design process progresses, with correspondingly smaller move limits for all design variables. The figure also demonstrates the versatility permitted by this scheme. Design variable 4 is initially permitted move limits of up to +/- 60%. As the variable became more important in reducing the objective function and satisfying the constraints, move limits were reduced to +/- 5%. The heuristics-based move limit strategy thus effectively replaced the involvement of the designer in making move limit choices.

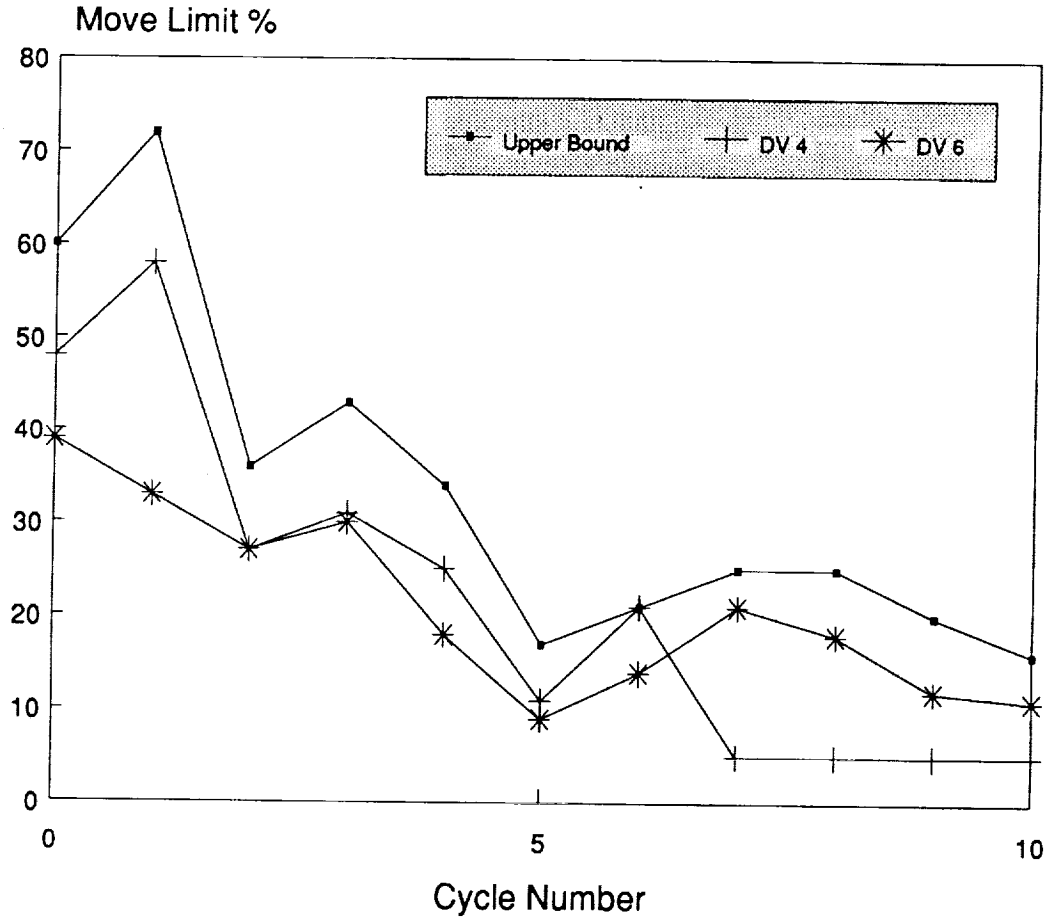


Figure 8.8 Variation of upper bound and move limits for selected design variables.

Table 8.4 Comparison of design variable allocations without (Case A) and with (Case B) heuristics.

	REALLOCATION COMPARISON			
CYCLE	DV1	DV5	DV7	DV8
Initial				
A	1	2	2	1
B	1	1	2	1
After 5				
A	1	1	1	1
B	1	1	2	1
After 10				
A	2	2	2	1
B	1	1	2	1
After 15				
A	1	1	2	1
B	1	1	1	2

WEIGHT AFTER 5 CYCLES

A	476 lb.	Without Heuristics
B	460 lb.	With Heuristics

The replacement of the coordination optimization problem with a heuristics-based coefficient assignment scheme was implemented. Table 8.5 shows the r and t coefficients corresponding to cumulative constraint 1 (C1) for the first ten cycles of the optimization process. At the initial cycle, no trade-off was permitted and the r coefficients were activated. At the first cycle, however, the constraints were sufficiently over-satisfied to permit a trade-off to occur. The trade-off continues until the 6th cycle, where the constraint exceeded the prescribed allowable limit and was determined to be active. The r coefficients were then active until such time as the constraint was satisfied for two consecutive cycles.

Table 8.5 Coefficient and constraint values for first ten optimization cycles.

Cycle	ACTIVE COEFFICIENTS				
	C1	R11	R12	T11	T12
0	-.20	.58	.42	-	-
1	-.13	-	-	-.1	.1
2	-.06	-	-	-.1	.1
3	-.02	-	-	-.1	.1
4	-.01	-	-	-.1	.1
5	-.006	-	-	-.1	.1
6	.009	.72	.28	-	-
7	.007	.71	.29	-	-
8	.003	.71	.29	-	-
9	.004	.71	.29	-	-
10	.0009	.09	.91	-	-

CHAPTER 9
CONCLUDING REMARKS

The recent thrust to improve quality of products as well as productivity in the United States has led to the realization that traditional design practices are inefficient and outmoded. It is in the design phase that the most potential exists to take advantage of the synergism of the inherently coupled disciplines to increase the overall quality of the product as well as to reduce the time and effort required for development and design of the product. The multidisciplinary interactions existing in large scale engineering design problems provide a unique set of difficulties associated with unwieldy numbers of design variables and constraints. Such obstacles require design techniques which take advantage of the extensive couplings of the disciplines in the analyses and optimizations, producing an efficient methodology to perform multidisciplinary synthesis.

The goal of the present effort was to develop a design capability appropriate for large engineering systems in which a distinct system hierarchy is difficult to identify. The concept of decoupling large complex problems into smaller, more tractable subsystems was investigated using system decomposition techniques. Three approaches to system decomposition were investigated for this purpose: the Global Sensitivity Equation (GSE) Method, the Concurrent Subspace Optimization (CSSO) Method, and the Concurrent Subspace Optimization - Embedded Expert System (CSSO-EES) Method. All three methods permit the concurrent consideration of the design criteria within all disciplines, thus providing an environment particularly amenable to parallel processing.

The GSE Method provides a means for decomposing the system into smaller subsystems that can be analyzed independently. Sensitivity information obtained by this approach is then used in an 'all-in-one' optimization, performed within the framework of a

sequential linearization strategy. The applicability of the GSE method in the multidisciplinary synthesis of aeronautical vehicles was investigated. A hypothetical aircraft in the class of the Cessna 170-type configuration was chosen as the test environment for the investigation, with the disciplines of structures, aerodynamics, and flight mechanics contributing to the objective function and constraints for the problem. Potential drawbacks in the use of the GSE approach were identified as arising from a large number of design variables and constraints and from an improper choice of design variables. Approximation methods were applied in order to reduce problem dimensionality and to improve the efficiency of the optimization process. The influence of constraint representations and the choice of design variables was shown to be a primary concern. Further, it was demonstrated that normalization of the system matrix is essential to avoid problems associated with ill-conditioning. Numerical results demonstrated the applicability of the GSE approach for large, coupled design synthesis problems.

The CSSO Method is basically an extension of the concept upon which the GSE is based. Not only are the analyses performed in separate subsystems, as in the GSE, but the optimizations are decoupled as well, and executed within separate subspaces concurrently. Unlike the conventional method of subspace optimizations, however, the CSSO eliminates the need for a full analysis in each subspace. Coordination amongst subspaces is achieved through the use of coordination coefficients which determine the responsibilities assigned to each subspace for satisfying a given constraint. Each cycle of the approach results in either a reduction of the system constraint violation or an improvement of the system objective function if the constraints are already satisfied. The temporarily decoupled optimization and analysis problems are eminently appropriate for a design organization setting in which groups of specialists are associated with disciplines and physical subsystems. Potential drawbacks were identified pertaining to the linearizations required throughout the CSSO. The synthesis problem investigated for method verification involved the optimal design of a ten-bar truss for minimum weight subject to stress and

displacement constraints. Subspaces were defined in terms of sizing and space variables. Results demonstrated the necessity for limiting movement of the coordination variables to achieve a smooth convergence. Further, the importance of imposing an appropriate approximation scheme was demonstrated. The applicability of a variable move limit strategy was shown to be of extreme computational worth. A verification study of the move limit strategy demonstrated the computational savings that can be obtained in the optimization process by permitting design variables to 'move' according to their impact on the design. Results of the CSSO verification study demonstrated that the approach was a versatile method which potentially offers exceptional data management advantages. The method allows for the use of specialized methods for analysis and optimization due to its modularity and is particularly suitable for the incorporation of human intervention and decision-making.

The heuristic variant of the CSSO, the CSSO-EES, takes advantage of problem-dependent heuristics and user expertise in the form of an embedded expert system capability to achieve improved convergence characteristics and greater versatility. The method makes use of heuristics in allocating the design variables to the most appropriate subspace, ensuring convergence within each approximate optimization problem, improving on the variable move limit strategy, and replacing the optimum sensitivity analysis and coordination optimization problem with the embedded expert system capability. The synthesis problem chosen for the demonstration involved the optimal design of a controls/structure interaction model for minimum weight based on active control and structural integrity requirements. Results demonstrated that improved efficiency as well as versatility can be obtained with the incorporation of an expert system capability.

Although the methods investigated proved applicable for the optimal design of large engineering systems, substantial room for improvement exists. The concept of multidisciplinary design optimization has gained prominent recognition in the engineering community in the last ten years. The importance of developing methodologies appropriate

for non-hierarchic environments has been emphasized as a key function of future design techniques. Such techniques require a necessary organizational as well as technical modification in order to achieve their full potential. The design process itself has undergone a major change, as the concept of incorporating all the design criteria in a simultaneous treatment has emerged as not only desirable, but necessary. Recent initiatives have focused on consideration of manufacturability and supportability as well as performance objectives in a 'Concurrent Engineering' (CE) approach. An investigation of the applicability of the CSSO and CSSO-EES in a CE problem is an obvious next step, as both methods allow for the use of specialized methods for analysis and optimization.

APPENDIX A
MOVE LIMIT STRATEGY VERIFICATION

Variable Move Limit Strategy

The move limit strategy defined in Chapter 7 is used to determine the move limits associated with each design variable as a result of their impact on the design. Side constraints are formulated for each design variable in terms of the move limits as,

$$\frac{(100-ml_i)}{100} * X_i \leq X_i \leq \frac{(100+ml_i)}{100} * X_i \qquad \text{(A1)}$$

This formulation defines upper and lower bounds associated with a prescribed percentage of movement about the design variables.

Method Implementation

The feasibility of the variable move limit strategy was investigated by means of a verification procedure. The method was applied in truss design problems with varying degrees of complexity and size. The design objectives and application models used in the verification procedure are described in the following sections.

Design Objectives

The objective of the design problem for all cases was to minimize the weight of the structure while maintaining limitations on member stresses and nodal displacements. The design variables were the cross-sectional member areas. The structural analysis was

performed using the finite element code EAL. The constrained minimization code, CONMIN, based on the method of usable-feasible directions, was used as the optimizer.

Application Models

Three test cases were investigated to demonstrate the feasibility of the move limit strategy in design optimization. Case 1 involved the optimal design of a ten-bar truss subject to static loading, as shown in Figure A1. Two initial designs were considered (1a and 1b), corresponding to initially infeasible and initially feasible points. Case 2 extended the implementation to a twenty-five-bar truss with the loadings shown in Figure A2. The final case involved the design of a two hundred-bar truss (Figure A3) with a 1000 lb. loading applied in the positive x direction at nodes 1,6,15,...,71. The allowable stress and displacement values for each case, as well as material property information are shown in Table A1.

Table A1 Material properties and allowable limits for move limit strategy verification applications.

Case	E(psi)	ρ(lbs/in^3)	σ_{al}(psi)	d_{al}(in)
1	10E6	0.100	+/- 25E3	+/- 2.0
2	10E6	0.100	+/- 25E3	+/- 2.0
3	30E6	0.283	+/- 10E3	+/- 0.5

Discussion of Results

Implementation of the variable move limit strategy in three test cases demonstrated significantly improved convergence characteristics. A discussion of results obtained for each test case is presented.

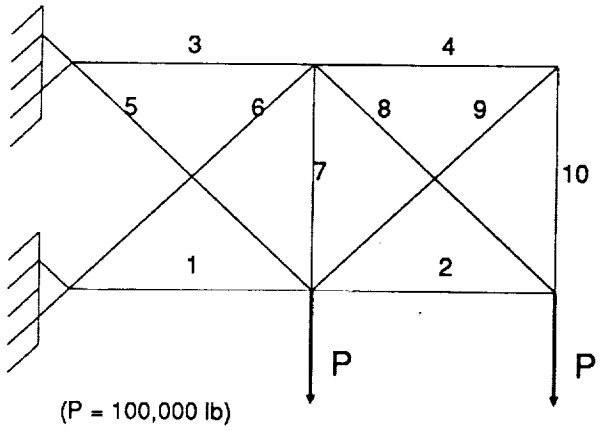


Figure A1 Case 1 model - 10 bar truss.

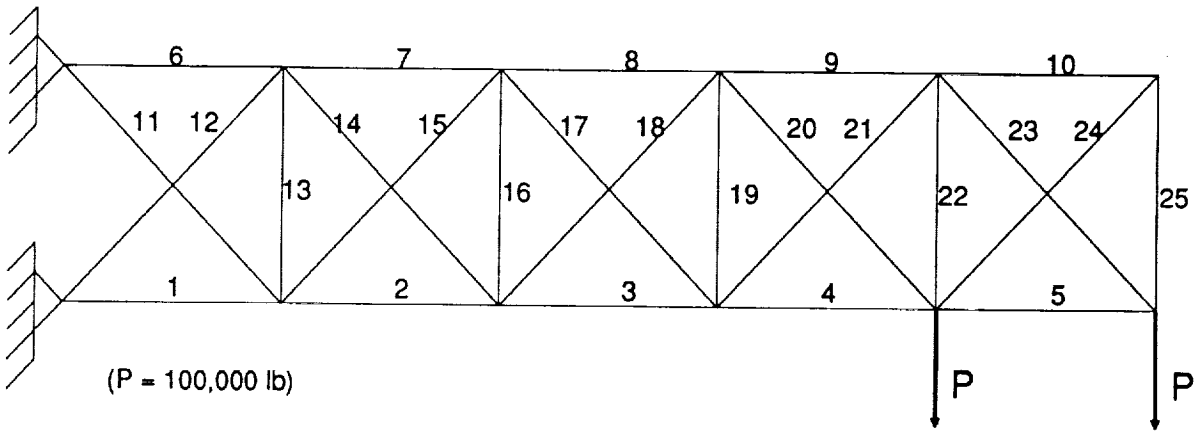


Figure A2 Case 2 model - 25 bar truss.

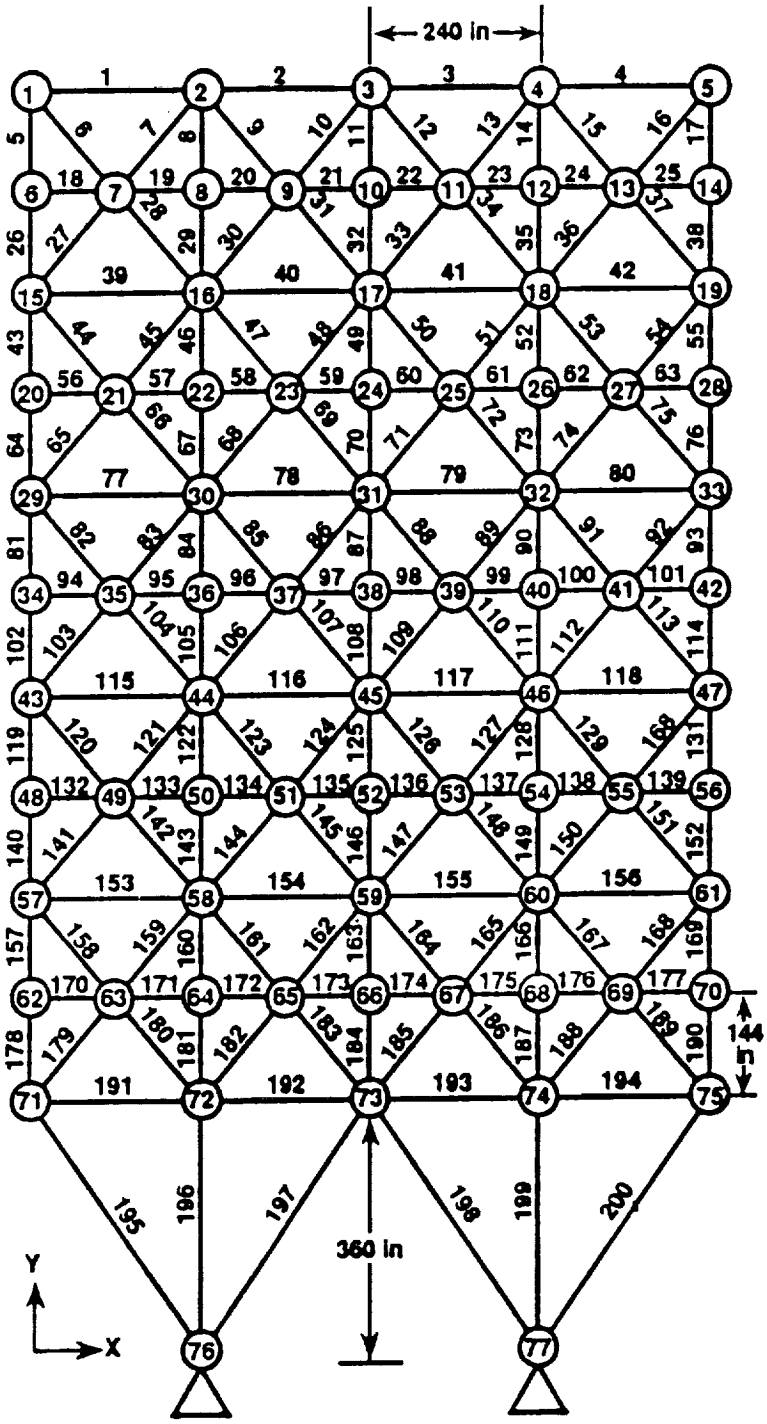


Figure A3 Case 3 model - 200 bar truss.

Case 1: Ten bar Truss

Two applications were investigated in this case, corresponding to initially infeasible (Case 1a) and initially feasible (Case 1b) design points. Figure A4 demonstrates the distribution of effectiveness coefficient values for Case 1a, in which the mean value of the effectiveness space and the lower and upper bounds are also shown. The convergence histories for the objective function and for a representative design variable are shown in Figures A5 and A6. Design variable 4 (DV 4) corresponds to the cross-sectional area of truss member 4 (Figure 3) and has a minimum gage value at the optimum. Figure A5 demonstrates the unacceptably large number of cycles required for DV 4 to approach the gage value, due to restrictive move limitations. The implementation of the move limit strategy permits the achievement of this value in only 8 cycles as opposed to the 32 cycles required without. Figure A7 demonstrates the move limit history for selected design variables over the ten cycles required for convergence. It can be seen from this figure that DV 4 consistently has move limits of 40% or greater throughout the optimization process, thus permitting a rapid convergence to the gage value.

A comparison of convergence histories for objective function and DV 4 is shown in Figure A8 for Case 1b. The first four cycles of the optimization process are shown to demonstrate the substantial improvement that is obtained in the first cycles by implementation of the move limit strategy. The objective function value was reduced by more than 36% with the addition of the move limit strategy as opposed to only 30% when it was not used. The change in DV 4 was most dramatic, however, with a 95% reduction in value as opposed to a 76% reduction.

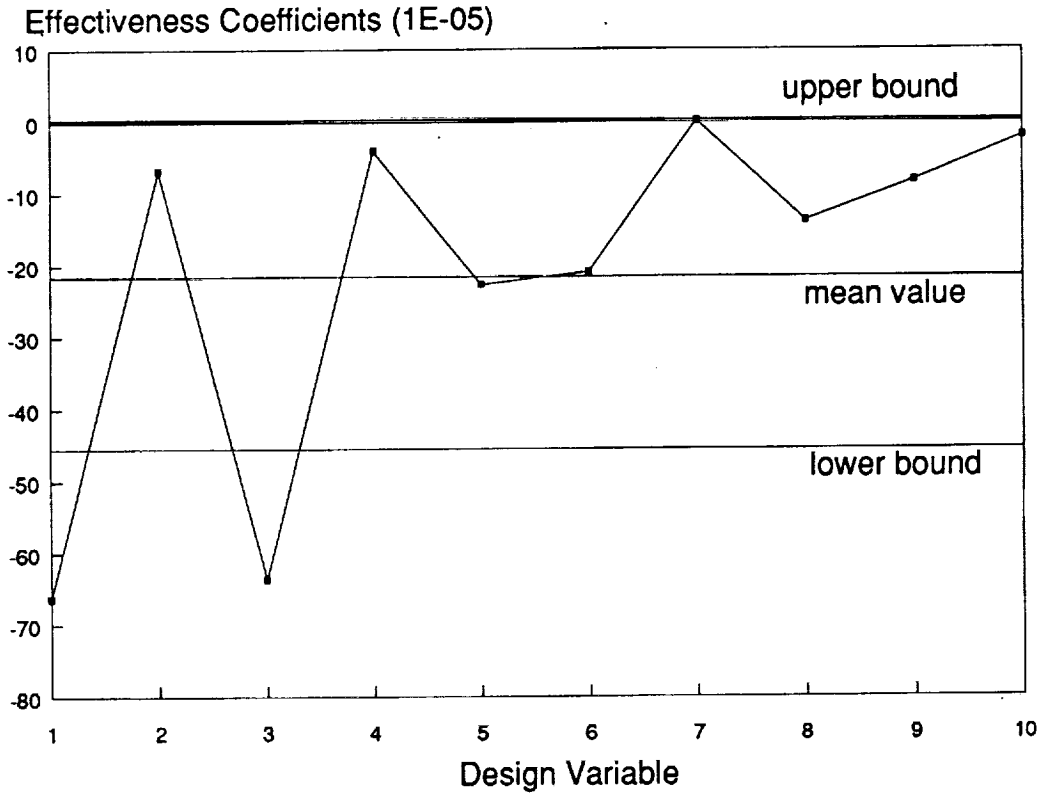


Figure A4 Effectiveness space for Case 1a initial point.

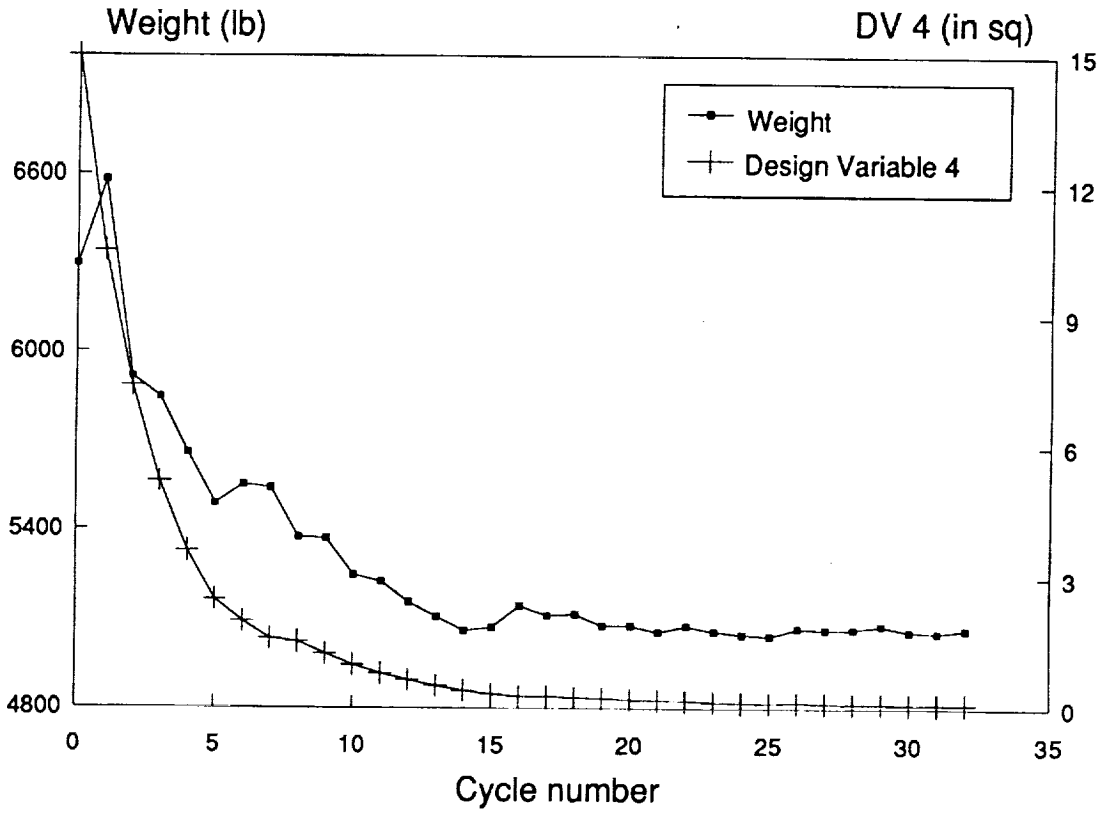


Figure A5 Case 1a convergence histories with no move limit strategy.

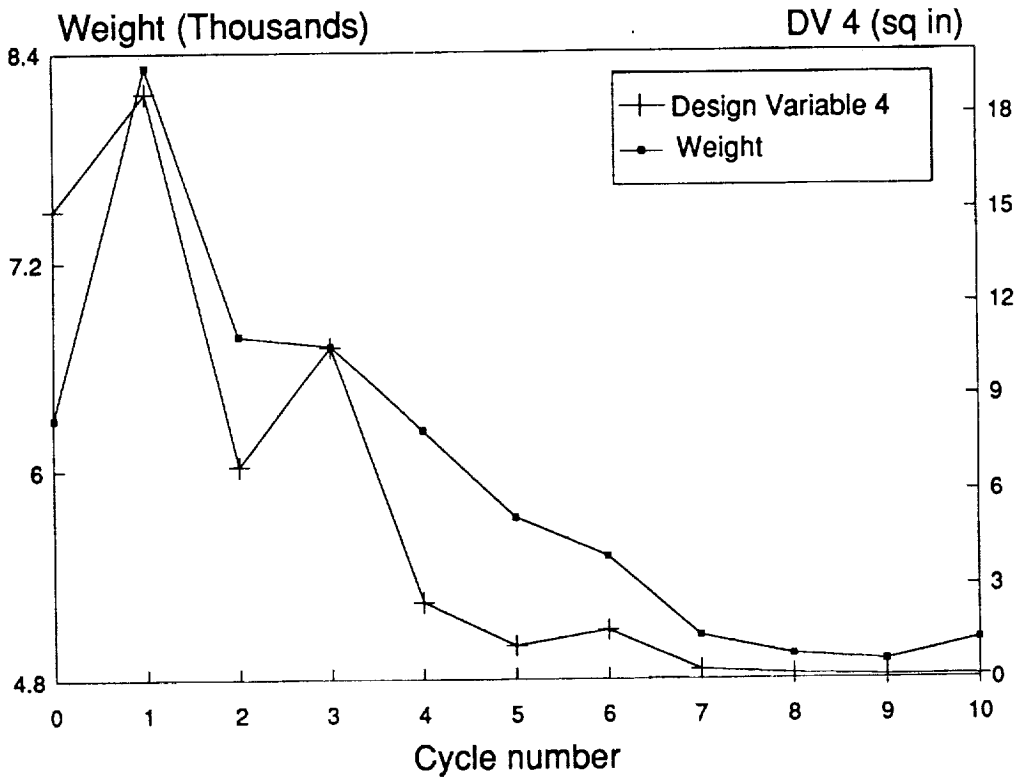


Figure A6 Case 1a convergence histories with move limit strategy.

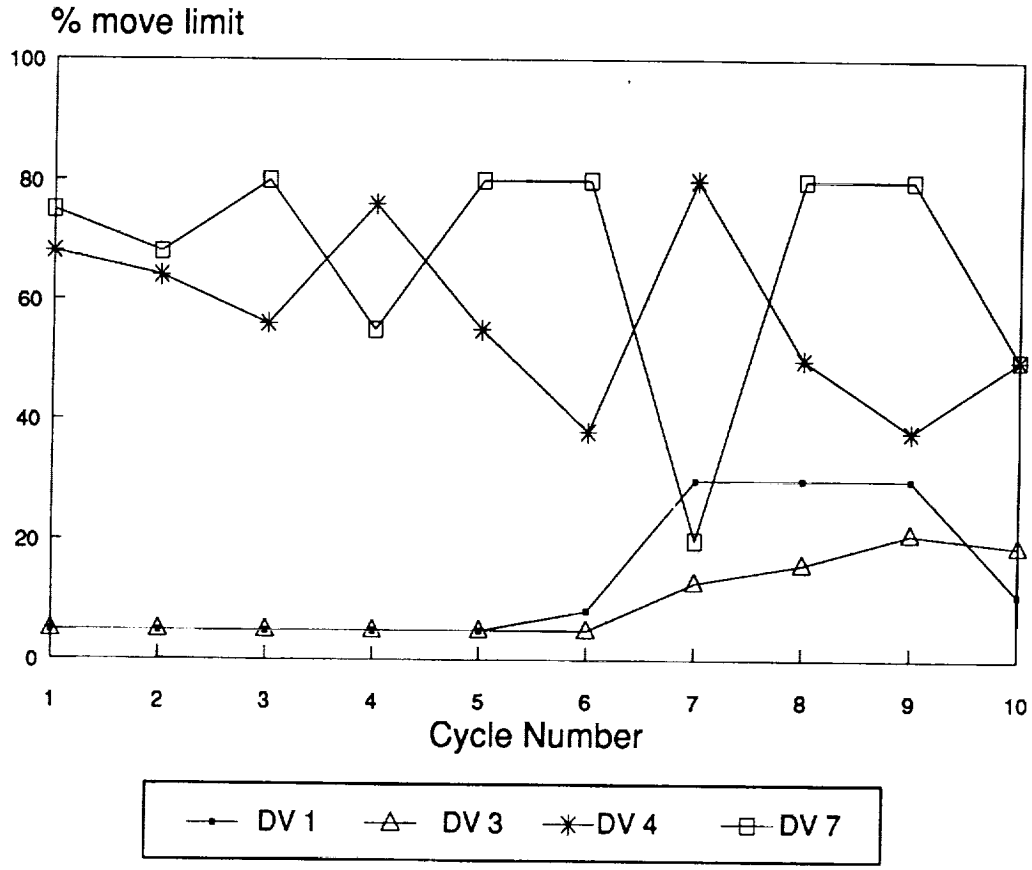


Figure A7 Move limit histories for Case 1a.

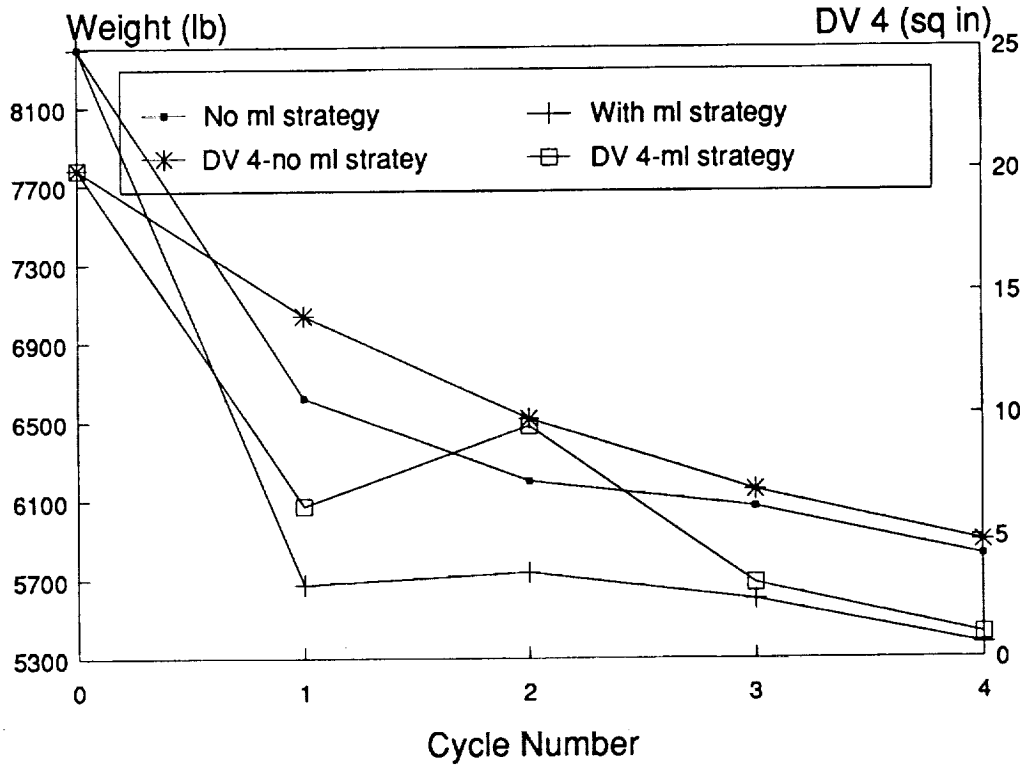


Figure A8 Comparison of Case 1b convergence histories with and without move limit strategy.

Case 2: Twenty-five bar Truss

The convergence histories for the objective function and for design variable 10 for Case 2 are shown in Figures A9 and A10. As with DV 4 in the ten-bar truss problem, DV 10 has a minimum gage value at the optimum. As with Case 1, an unacceptably large number of cycles are again required for DV 10 to approach the gage value, due to the restrictive move limitations applied in the design process when no efficient move limit strategy exists (Figure A9). The implementation of the move limit strategy allows DV 10 to reach gage value in only 12 cycles as opposed to more than 40 cycles required with no move limit strategy.

Figures A9 and A10 demonstrate that a 59% difference exists in the computational requirements for the two cases, with the move limit strategy case requiring only 17 cycles for convergence and the other requiring more than 42. As with Case 1, it was demonstrated that implementation of the move limit strategy results in greatly improved convergence characteristics, which translates into computational savings.

Case 3: Two-hundred bar Truss

Application of the move limit strategy in the minimization of weight for the two hundred bar truss example was made for one cycle to demonstrate the savings that can potentially be obtained in such a large problem. The objective function values for the initial and first cycle are shown in Table A2.

Table A2 Initial and first cycle results for two-hundred bar truss.

Cycle	Weight (lb)	
	no ml strategy	with ml strategy
0	149451	149451
1	119585	80107

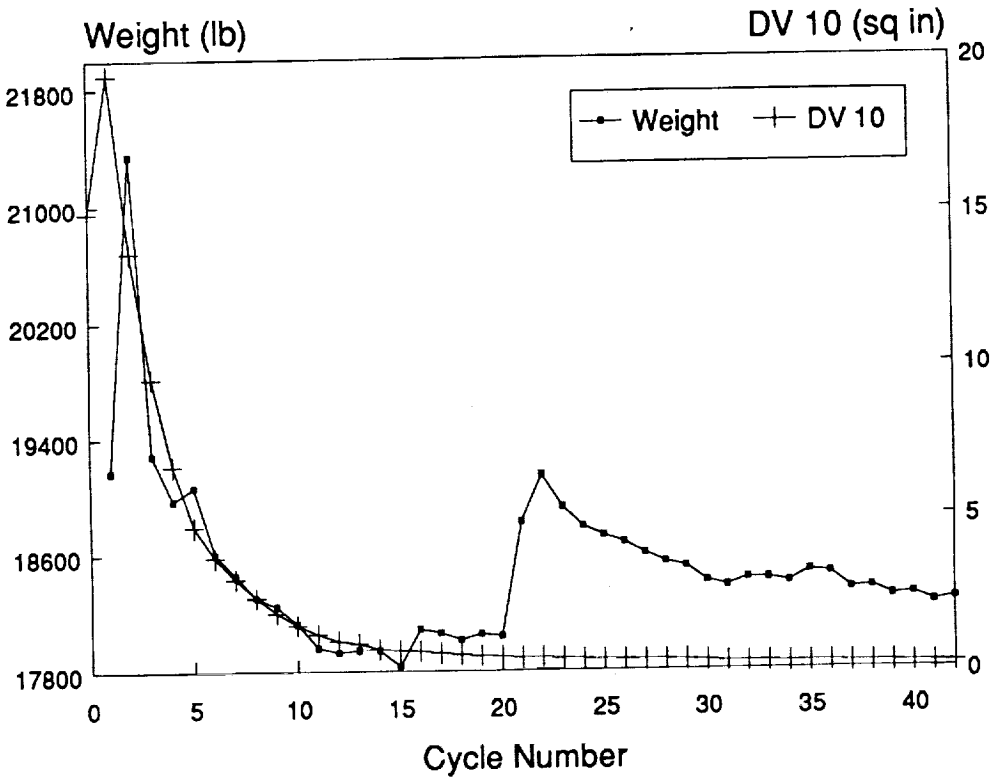


Figure A9 Case 2 convergence histories with no move limit strategy.

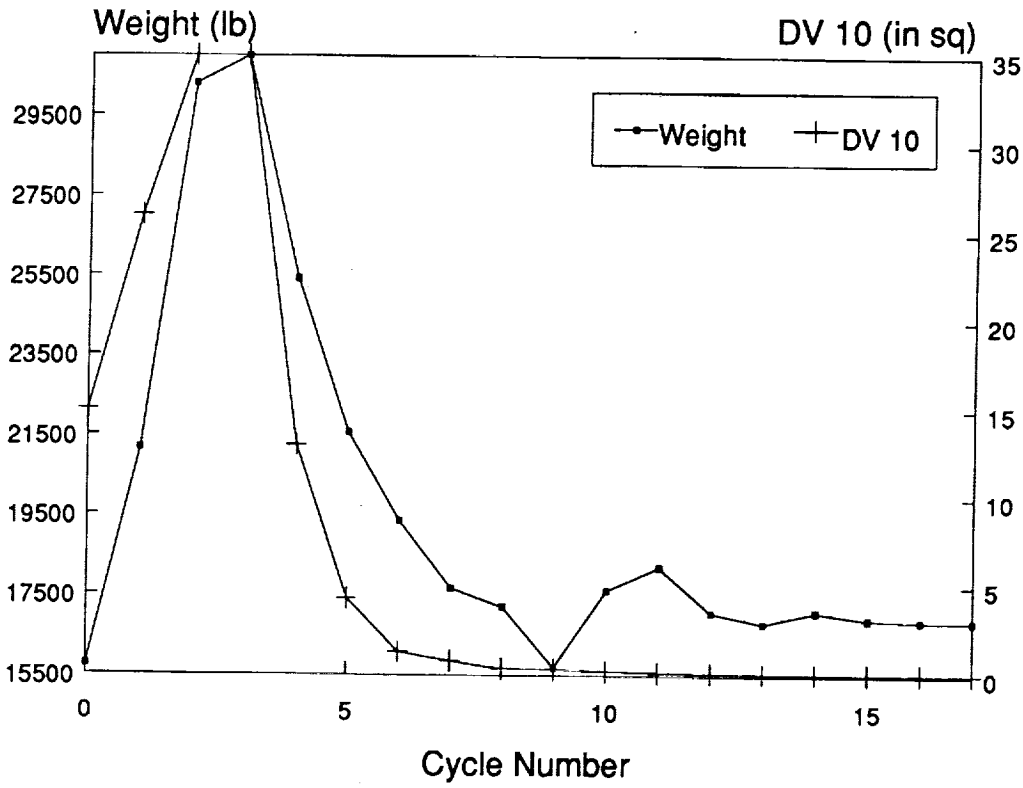


Figure A10 Case 2 convergence histories with move limit strategy.

A 46% improvement is made in objective function value with the move limit strategy as opposed to only a 20% improvement without, with no constraint violations in either case.

Concluding Remarks for Move Limit Strategy Verification

The applications of a variable move limit strategy in the optimal design process were demonstrated. The strategy is based on effectiveness coefficients which quantify a design variable's impact on the design criteria. Results for three test cases of varying size and complexity demonstrate substantial reductions in computational effort, which translates into increased efficiency. It has been shown that the ability exists to take the guesswork out of move limit value assignment.

APPENDIX B
KNOWLEDGE BASE FOR CONCURRENT SUBSPACE OPTIMIZATION - EMBEDDED EXPERT SYSTEM METHOD

Design Variable Allocation

```
;CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
;
;       DVRL.CLP
;
;       DESIGN VARIABLE ALLOCATION RULE SET
;
;CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
;
;       ALLOCATION RULE
;
;       Determines total design variables allocated. If number design variables allocated
;       is less than total number design variables, then allocation is unclear.
;
;(defrule allocation
        (ndv_structures_allocated       ?ndvsall)
        (ndv_controls_allocated         ?ndvcall)
        (ndv_total                      ?ndvtot)
=>
        (bind   ?ndv_allocate  (+       ?ndvsall        ?ndvcall))
        (if     (<      ?ndv_allocate  ?ndvtot)
        then    (assert (allocation     unclear))))
;
;
;CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
;
;       STRUCTURES_NDV_SIZE RULE
;
;       Determines whether number design variables allocated to structures is large
;       or not large.
;
(defrule structures_ndv_size
        (allocation                     unclear)
        (ndv_structures_allocated       ?ndvsall)
        (ndv_total                      ?ndvtot)
=>
        (bind   ?newtotal       (*      .6      ?ndvtot))
        (if     (<      ?ndvsall        ?newtotal)
        then    (assert (ndv_structures         not_large))
        else    (assert (ndv_structures         large))))
```

```
;CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
;
;       CONTROLS_NDV_SIZE RULE
;
;       Determines whether number design variables allocated to controls is large
;       or not large.
;
(defrule controls_ndv_size
        (allocation                 unclear)
        (ndv_controls_allocated     ?ndvcall)
        (ndv_total                  ?ndvtot)
=>
        (bind  ?newtotal   (*    .6      ?ndvtot))
        (if    (<    ?ndvcall         ?newtotal)
        then   (assert (ndv_controls  not_large))
        else   (assert (ndv_controls  large))))
;
;
;CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
;
;       ALLOCATE_SIZING RULE
;
;       Allocates sizing variable to either structures or controls based on numbers of
;       design variables already allocated as well as traditional considerations.
;
(defrule allocate_sizing
        (allocation           unclear)
        (dv                   sizing)
        (ndv_structures       ?ndvs)
=>
        (if     (eq     ?ndvs  large)
        then    (bind   ?alldv controls)
        else    (bind   ?alldv structures))
        (KBANS1    ALLOCATE   ?alldv    0     0))
;
;
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
;
;       ALLOCATE_DAMPING RULE
;
;       Allocates damping variable to either structures or controls based on numbers of
;       design variables already allocated as well as traditional considerations.
;
(defrule allocate_damping
        (allocation           unclear)
        (dv                   damping)
        (ndv_controls         ?ndvc)
=>
        (if     (eq     ?ndvc  large)
        then    (bind   ?alldv structures)
        else    (bind   ?alldv controls))
        (KBANS1    ALLOCATE   ?alldv    0     0))
```

Variable Move Limit Strategy

```
;CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
;
;       ML.CLP
;
;       RULE SET FOR VARIABLE MOVE LIMIT DETERMINATION
;
;CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
;
;       CONSTRAINT RULE
;
;       Determines whether constraint is active, not-active, or violated.
;
(defrule constraint_satisfaction
        (constraint             ?g)
        (constraint_thickness   ?ct)
=>
        (bind   ?ctneg (-       0.0     ?ct))
        (if     (<      ?g      ?ctneg)
        then    (assert (satisfaction   not_active))
        else    (if     (<      ?g      ?ct)
                then    (assert (satisfaction   active))
                else    (assert (satisfaction   violated)))))
;
;
;CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
;
;       INITIALIZE RULE
;
;       If first cycle, set upper bound for move limits.
;
(defrule initial
        (cycle                  ?cyc)
        (upper_bound            ?ub)
=>
        (if     (=      ?cyc    0)
        then    (KBANS1     UPPER       null    ?ub     0)))
;
;
;CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
;
;       ACTIVE RULE
;
;       If cycle greater than zero and constraint is active, determine upper bound based
;       on whether constraint is positive or negative.
;
(defrule active_constraint
        (satisfaction           ?active)
        (cycle                  ?cyc)
        (constraint             ?g)
        (upper_bound            ?ub)
=>
```

```
        (if     (>      ?cyc    0)
        then    (if     (<      ?g      0.0)
                then    (bind   ?newub          ?ub)
                else    (bind   ?newub          (*      0.9     ?ub)))
                (KBANS1         UPPER           null    ?newub          0)))
;
;
;CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
;
;       NOT_ACTIVE RULE
;
;       If cycle greater than zero and constraint not active then determine upper bound
;       based on how close to allowable upper limit previous cycle was.
;
(defrule not_active_constraint
        (satisfaction           not_active)
        (cycle                  ?cyc)
        (upper_bound            ?ub)
        (upper_max              ?ubm)
=>
        (if     (>      ?cyc    0)
        then    (bind   ?ubmax          (*      0.83333         ?ubm))
                (if     (<      ?ub     ?ubmax)
                then    (bind   ?newub          (*      1.2     ?ub))
                else    (bind   ?newub          ?ubm))
                (KBANS1         UPPER           null    ?newub          0)))
;
;
;CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
;
;       VIOLATED RULE
;
;       If cycle greater than zero and constraint is violated, then decreased upper
;       bound by 70%.
;
(defrule violated_constraint
        (satisfaction           violated)
        (cycle                  ?cyc)
        (upper_bound            ?ub)
=>
        (bind   ?newub          (*      0.7     ?ub))
        (KBANS1         UPPER           null    ?newub          0))
```

Optimization Parameter Determination

```
;CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
;
;       OPRL.CLP
;
;       OPTIMIZATION PARAMETER RULE SET FOR PREMATURE
;       CONVERGENCE
;
;CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
;
;       STATUS RULE
;
;       Determines whether termination due to satisfaction of convergence criteria or
;       not.
;
(defrule status
                (declare        (salience       500))
        ?n1 <-  (fire           1)
                (iter           ?it)
                (itmax          ?itm)
=>
        (retract        ?n1)
        (if     (=      ?it     ?itm)
        then    (assert iter_status     equal))
        else    (assert iter_status     not_equal))))
;
;
;CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
;
;       FEASIBLE SOLUTION RULE
;
;       If termination premature and solution at final iteration is feasible, increases
;       allowable number of iterations until upper bound is exceeded.
;
(defrule solution_feasible
                (declare        (salience       300))
                (iter_status    equal)
                (solution       feasible)
        ?n1 <-  (itmax          ?itm)
                (itmax_bound    ?itmbound)
        ?n2 <-  (fire           2)
=>
        (retract        ?n2)
        (bind   ?newitmax       (*      2.      ?itm))
        (if     (<=     ?newitmax       ?itmbound)
        then    (retract        ?n1)
                (assert         (itmax  ?newitmax)
        else    (assert         (itmax_bound_is         exceeded))))
```

```
;CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
;
;       INFEASIBLE SOLUTION 1 RULE
;
;       If termination premature, solution at final iteration is infeasible, and phi has
;       not yet been adjusted, increases value of phi.
;
(defrule solution_not_feasible_1
                (declare        (salience       300))
                (iter_status    equal)
                (solution       not_feasible)
        ?n1 <-  (phi            ?ph)
                (phi_init       ?phin)
        ?n2 <-  (fire           3)
=>
        (retract        ?n2)
        (if     (=      ?ph     ?phin)
        then    (bind   ?newphi         (*      3.      ?ph))
                (retract        ?n1)
                (assert         (phi    ?newphi))))
;
;
;CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
;
;       INFEASIBLE SOLUTION 2A RULE
;
;       If termination premature, solution at final iteration is infeasible, and phi has
;       already been adjusted, increase value of phi as long as bound not exceeded.
;
(defrule solution_not_feasible_2a
                (declare                (salience       300))
                (iter_status            equal)
                (solution               not_feasible)
        ?n1 <-  (phi                    ?ph)
                (phi_init               ?phin)
                (phi_bound              ?phbound)
                (phi_increment          ?phincrem)
        ?n2 <-  (itmax                  ?itm)
                (itmax_bound            ?itbound)
        ?n3 <-  (fire                   4)
=>
        (retract        ?n3)
        (if     (>      ?ph     ?phin)
        then    (bind   ?newphi         (+      ?phincrem       ?ph))
                (bind   ?itmnew         (*      ?2.     ?itm))
                (if     (<=     ?newphi         ?phbound)
                then    (if     (<=     ?itmnew         ?itbound)
                        then    (retract        ?n1     ?n2)
                                (assert         (phi    ?newphi)
                                (assert         (itmax  ?itmnew))))))
```

```
;CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
;
;       INFEASIBLE SOLUTION 2B RULE
;
;       If termination premature, solution at final iteration is infeasible, phi has
;       already been adjusted, and value of phi bound has been exceeded, assert such
;       into fact-list.
;
(defrule solution_not_feasible_2b
                (declare            (salience       300))
                (iter_status        equal)
                (solution           not_feasible)
                (phi                ?ph)
                (phi_init           ?phin)
                (phi_bound          ?phbound)
                (phi_increment      ?phincrem)
        ?n1 <-  (fire               5)
=>
        (retract        ?n1)
        (if     (>      ?ph     ?phin)
        then    (bind   ?newphi         (+      ?phincrem       ?ph))
                (bind   ?itmnew         (*      2.      ?itm))
                (if     (>      ?newphi         ?phbound)
                then    (assert         (phi_bound_is_exceeded)))))
;
;
;CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
;
;       INFEASIBLE SOLUTION 2C RULE
;
;       If termination premature, solution at final iteration is infeasible, phi has
;       already been adjusted, and value of itmax bound has been exceeded, assert
;       such into fact-list.
;
(defrule solution_not_feasible_2b
                (declare            (salience       300))
                (iter_status        equal)
                (solution           not_feasible)
        ?n1 <-  (phi                ?ph)
                (phi_init           ?phin)
        ?n2 <-  (itmax              ?itm)
                (itmax_bound        ?itbound)
        ?n3 <-  (fire               6)
=>
        (retract        ?n3)
        (if     (>      ?ph     ?phin)
        then    (bind   ?itmnew         (*      2.      ?itm))
                (if     (>      ?itmnew         ?itbound)
                then    (assert         (itmax_bound_is_exceeded)))))
```

```
;CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
;
;       ITMAX EXCEEDED RULE
;
;       If itmax bound is exceeded and solution feasible, adjust convergence parameters.
;       If solution infeasible, adjust parameters associated with constraint satisfaction.
;       If parameter bounds exceeded, assert 'process stop' into fact-list.
;
(defrule itmax_exceeded
                (declare              (salience      100))
                (itmax_bound_is       exceeded)
        ?n1 <-  (delfun               ?delf)
                (delfun_increment     ?delfincrem)
        ?n2 <-  (dabfun_variable      ?dabvar)
                (dabfun_increment     ?dabincrem)
        ?n3 <-  (itrm                 ?itr)
                (new_itrm             ?newitrm)
        ?n4 <-  (theta                ?theta)
        ?n5 <-  (ct                   ?ctv)
                (ct_increment         ?ctincrem)
                (solution             ?sol)
                (delfun_bound         ?delbound)
                (dabvar_bound         ?dabbound)
                (theta_bound          ?thetabound)
                (ct_bound             ?ctbound)
        ?n6 <-  (fire 7)
=>
        (retract        ?n6)
        (if     (eq     ?sol    feasible)
        then    (bind   ?newdelfun    (+      ?delf           ?delfincrem))
                (bind   ?newdabvar    (+      ?dabvar         ?dabincrem))
                (if     (<=     ?newdelfun    ?delbound)
                then    (retract        ?n1     ?n2)
                        (assert         (delfun                 ?newdelfun))
                        (assert         (dabfun_variable        ?newdabvar)))
                else    (if     (>      ?itr    ?newitrm)
                        then    (retract        ?n3)
                                (assert         (itrm   ?newitrm))
                        else    (assert         (process        stop))))
        else    (bind   ?newthet        (*      2.      ?thet))
                (bind   ?newct          (-      ?ctv    ?ctincrem))
                (if     (<=     ?newthet        ?thetabound)
                then    (if     (>=     ?newct          ?ctbound)
                        then    (retract        ?n4     ?n5)
                                (assert         (theta  ?newthet))
                                (assert         (ct     ?newct)))
                else    (assert (process        stop)))))
```

```
;CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
;
;       PHI EXCEEDED RULE
;
;       If phi bound exceeded and solution infeasible, adjust parameters associated
;       with constraint satisfaction. If parameter bounds exceeded, assert 'process
;       stop' into fact-list.
;
(defrule phi_exceeded
                (declare            (salience       100))
                (phi_bound_is       exceeded)
        ?n1 <-  (theta              ?thet)
        ?n2 <-  (ct                 ?ctv)
                (ct_increment       ?ctincrem)
                (solution           ?sol)
                (theta_bound        ?thetabound)
                (ct_bount           ?ctbound)
        ?n3 <-  (fire               8)
=>
        (retract        ?n3)
        (if     (eq     ?sol    not_feasible)
        then    (bind   ?newthet        (*      2.      ?thet))
                (bind   ?newct          (-      ?ctv    ?ctincrem))
                (if     (<      ?newthet        ?thetabound)
                then    (if     (<      ?newct          ?ctbound)
                        then    (retract        ?n1     ?n2)
                                (assert         (theta  ?newthet))
                                (assert         (ct     ?newct)))
                        else    (assert         (process        stop)))))
;
;
;CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
;
;       PRINT RULE
;
;       Subroutine KBANS1 used to update new parameter values.
;
(defrule parameters_out
                (declare            (salience       10))
                (iter_status        equal)
        ?n1 <-  (itmax              ?itm)
                (phi                ?ph)
                (delfun             ?delf)
                (dabfun_variable    ?dabvar)
                (itrm               ?itr)
                (theta              ?the)
                (ct                 ?ctv)
=>
        (retract        ?n1)
        (KBANS1         ITMAX           ITRM            ?itm            ?itr)
        (KBANS1         DELFUN          DABVAR          ?delf           ?dabvar)
        (KBANS1         PHI             THETA           ?ph             ?the)
        (KBANS1         CT              null            ?ctv            0))
```

```
;CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
;
;       PRINT STOP RULE
;
;       If 'process stop' asserted into fact-list at any time, KBANS1 used to relay
;       information to main routine.
;
(defrule process_stop
            (declare            (salience       10))
        ?n1 <-  (process        stop)
=>
        (retract            ?n1)
        (KBANS1     STOP        null        0           0))
```

Coordination Coefficient Assignment

```
;CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
;
;       COEFRL.CLP
;
;       COEFFICIENT ASSIGNMENT RULE SET
;
;CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
;
;       INITIALIZE RULE
;
;       If first cycle, set switch parameters.
;
(defrule initial
                (declare                (salience       500))
                (cycle                  ?cyc)
        ?n1 <-  (fire                   1)
=>
        (retract        ?n1)
        (if     (=      ?cyc    0)
        then    (assert (process        stop))
                (bind   ?s1     1)
                (bind   ?s2     1)
                (KBANS1     INS1    INS2   ?s1      ?s2)
        else    (assert (process        no_stop))))
;
;
;CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
;
;       CONSTRAINT1A RULE
;
;       Determines constraint satisfaction for present and previous cycles.
;
(defrule constraint1a_satisfaction
                (declare                (salience       300))
                (process                no_stop)
        ?n1 <-  (g1_present             ?g1)
        ?n2 <-  (g1_past                ?g1old)
                (constraint_thickness   ?ct)
=>
        (bind   ?ctneg  (-      0.0     ?ct))
        (if     (>=     ?g1     ?ctneg)
        then    (bind   ?s1     1)
                (retract        ?n1)
                (KBANS1     S1      null    ?s1     0)
        else    (if     (>=     ?g1old          ?ctneg)
                then    (bind   ?s1     1)
                        (KBANS1     S1      null    ?s1     0)
                        (retract        ?n2))))
```

```
;CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
;
;       CONSTRAINT2A RULE
;
;       Determines constraint satisfaction for present and previous cycles.
;
(defrule constraint2a_satisfaction
                (declare                (salience       100))
                (process                no_stop)
        ?n1 <- (g2_present              ?g2)
        ?n2 <- (g2_past                 ?g2old)
                (constraint_thickness   ?ct)
=>
        (bind   ?ctneg  (-      0.0     ?ct))
        (if     (>=     ?g2     ?ctneg)
        then    (bind   ?s2     1)
                (retract        ?n1)
                (KBANS1         S2      null    ?s2     0)
        else    (if     (>=     ?g2old          ?ctneg)
                then    (bind   ?s1     1)
                        (KBANS1         S2      null    ?s2     0)
                        (retract        ?n2))))
;
;
;CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
;
;       CONSTRAINT1B RULE
;
;       Determines constraint satisfaction for present and previous cycles.
;
(defrule constraint1b_satisfaction
                (declare                (salience       300))
                (process                no_stop)
                (g1_present             ?g1)
                (g1_past                ?g1old)
                (constraint_thickness   ?ct)
        ?n1 <- (fire                    2)
=>
        (bind   ?ctneg  (-      0.0     ?ct))
        (if     (<      ?g1     ?ctneg)
        then    (if     (<      ?g1old          ?ctneg)
                then    (bind   ?s1     0)
                        (assert (status1        trade-off))
                        (KBANS1         S1      null    ?s1     0)
                        (retract        ?n1))))
```

```
;CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
;
;       CONSTRAINT2B RULE
;
;       Determines constraint satisfaction for present and previous cycles.
;
(defrule constraint2b_satisfaction
                (declare                (salience       100))
                (process                no_stop)
                (g2_present             ?g2)
                (g2_past                ?g2old)
                (constraint_thickness   ?ct)
        ?n1 <-  (fire                   3)
=>
        (bind   ?ctneg  (-      0.0     ?ct))
        (if     (<      ?g2     ?ctneg)
        then    (if     (<      ?g2old          ?ctneg)
                then    (bind   ?s2     0)
                        (assert (status1        trade-off))
                        (KBANS1 S2      null    ?s2     0)
                        (retract        ?n1))))
;
;
;CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
;
;       TRADE-OFF1 RULE
;
;       Determines t coefficient values.
;
(defrule trade1
                (declare                (salience       10))
                (process                no_stop)
        ?n1 <-  (status1                trade-off)
                (g1_present             ?g1)
                (g1_past                ?g1old)
                (obj_gradient1          ?dwdx1)
                (obj_gradient2          ?dwdx2)
=>
        (retract        ?n1)
        (if     (>      ?dwdx1          dwdx2)
        then    (if     (>      ?g1     ?g1old)
                then    (bind   ?t11    -.1)
                        (bind   ?t12    .1)
                else    (bind   ?t11    -.2)
                        (bind   ?t12    .2))
        else    (if     (?      ?g1     ?g1old)
                then    (bind   ?t11    .1)
                        (bind   ?t12    -.1)
                else    (bind   ?t11    .2)
                        (bind   ?t12    -.2)))
        (KBANS1 T11     T12     ?t11    ?t12))
```

```
;CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
;
;       TRADE-OFF2 RULE
;
;       Determines t coefficient values.
;
(defrule trade2
                (declare              (salience     10))
                (process              no_stop)
        ?n1 <-  (status2              trade-off)
                (g_present            ?g2)
                (g2_past              ?g2old)
                (obj_gradient1        ?dwdx1)
                (obj_gradient2        ?dwdx2)
=>
        (retract        ?n1)
        (if     (>      ?dwdx1          dwdx2)
        then    (if     (>      ?g2     ?g2old)
                then    (bind   ?t21    -.1)
                        (bind   ?t22    .1)
                else    (bind   ?t21    -.2)
                        (bind   ?t22    .2))
        else    (if     (?      ?g2     ?g2old)
                then    (bind   ?t21    .1)
                        (bind   ?t22    -.1)
                else    (bind   ?t21    .2)
                        (bind   ?t22    -.2)))
        (KBANS1 T21     T22     ?t21    ?t22))
```

REFERENCES

Arm78 Armstrong, E. S., "ORACLS–A System for Linear-Quadratic-Gaussian Control Law Design," NASA TP-1106, 1978.

Bar88 Barthelemy, J.-F. M., and Riley, M. F., "Improved Multilevel Optimization Approach for the Design of Complex Engineering Systems", AIAA Journal, Vol. 26, No. 3, March 1988, pp. 353-360.

Blo87 Bloebaum, C. L., "Implementation of Global Sensitivity Analysis in Structural/Control Optimization", Master's Thesis, University of Florida, 1987.

Bry69 Bryson, A. E., Ho, Y. C., Applied Optimal Control (Blaisdell Publishing Co., 1969, Waltham, Mass.), Chapter 5, p.152.

Dhi84 Dhir, S. K. and Hurwitz, M. M., "Role of Optimization in Interdisciplinary Analyses of Naval Structures", Recent Experiences in Multidisciplinary Analysis and Optimization, NASA CP-2327, 1984.

Don79 Dongarra, J.J.; Bunch, J. R.; Moler, C. B.: and Steward, G. W.: LINPACK User's Guide. SIAM, 1979.

Etk82 Etkin, B., Dynamics of Flight Stability and Control (John Wiley & Sons, 1982, New York, N.Y.).

Fad90 Fadel,G. M., Riley, M. F., Barthelemy, J.-F., "Two point Exponential Approximation Method for Structural Optimization", Structural Optimization, Vol. 2, 1990, pp. 117-124.

Fer84 Ferebee, M. J., "IDEAS, A Multidisciplinary Computer-Aided Conceptual Design System for Spacecraft", Recent Experiences in Multidisciplinary Analysis and Optimization, NASA CP-2327, 1984.

Fou54 Foulkes, J., "The Minimum Weight Design of Structural Frames", Proceedings of the Royal Society (London), Vol. A223, No. 1155, 1954.

Ful73 Fulton, R. E., Sobieszczanski, J. E., Storaasli, O., Landrum, E. J., and Leondorf, D., "Application of Computer-Aided Aircraft Design in a Multidisciplinary Environment", AIAA Paper 73-353, March 1973.

Gel66 Gellatly, R. A., "Development of Procedures for Large Scale Automated Minimum Weight Structural Design", AFFDL-TR-66-180, 1966.

Gel71 Gellatly, R. A., Berke, L. and Gibson, W., "The Use of Optimality Criteria in Automated Structural Design", Presented at the 3rd Conference on Matrix Methods in Structural Mechanics, WPAFB, Ohio, Oct. 1971.

Ger56 Gerard, G., Minimum Weight Analysis of Compressive Structures (New York University Press, 1956, New York, N.Y.).

Gia89 Giarratano, J.C., CLIPS User's Guide, Lyndon B. Johnson Space Center, June 1989.

Gil72 Giles, G. L., Blackburn, C. L., and Dixon, S. C., "Automated Procedures for Sizing Aerospace Vehicle Structures (SAVES)", Journal of Aircraft, Vol. 9, Dec. 1972, pp. 812-819.

Gri61 Griffith, R.E., Stewart, R.A. , A Nonlinear Programming Technique for the Optimization of Continuous Processing Systems, Management Science, Vol. 7, 1961, pp. 379-392.

Haf76 Haftka, R. T. and Starnes, J. H., "Applications of a Quadratic Extended Interior Penalty Function for Structural Optimization", AIAA Journal, Vol. 14, June 1976, pp. 718-724.

Haf80 Haftka, R., "Sensitivity Analysis", Proceedings of the NASA-ASI Session on Modern Structural Optimization, Liege, Belgium, August 1980.

Haf90 Haftka, R. T., Gurdal, Z. and Kamat, M. P., Elements of Structural Optimization (Kluwer Academic Publishers, 1990, Boston, Mass.).

Haj81 Hajela, P., Sobieszczanski-Sobieski, J., "The Controlled Growth Method - A Tool for Structural Optimization", Proceedings of the AIAA/ASME/ASCE/AHS 22nd Structures, Structural Dynamics, and Materials Conference, 1981.

Haj82 Hajela, P., Techniques in Optimum Structural Synthesis with Static and Dynamic Constraints, Ph.D. Dissertation, Stanford University, June 1982.

Hey51 Heyman, J., "Plastic Design of Beams and Frames for Minimum Material Consumption", Quarterly of Applied Mathematics, Vol. 8, 1951, pp. 373-380.

Hug84 Hughes, O. F., "Structural Optimization of Large Ocean-Going Structures", Recent Experiences in Multidisciplinary Analysis and Optimization, NASA CP-2327, 1984.

Kar68 Karnes, R. N. and Tocher, J. L., "Automated Design of Optimum Hole Reinforcement", Boeing Report D6-23359, 1968.

Kic68 Kicher, T. P., "Structural Synthesis of Integrally Stiffened Cylinders", Journal of Spacecraft and Rockets, Vol. 5, Jan. 1968, pp. 62-67.

Kle55 Klein, B., "Direct Use of External Principles in Solving Certain Optimization Problems Involving Inequalities", Journal of the Operations Research Society of America, Vol. 3, 1955, pp. 345-346.

Kro88 Kroo, I. and Takai, M., "A Quasi-Procedural, Knowledge-Based System for Aircraft Design", AIAA/AHS/ASEE Aircraft Design, Systems, and Operations Meeting, Atlanta Georgia, Sept. 1988, AIAA Paper No. 88-4428.

Lem84 Lemmerman, L. A., "Optimization in the Systems Engineering Process", Recent Experiences in Multidisciplinary Analysis and Optimization, NASA CP-2327, 1984.

Liv56 Livesley, R. K., "The Automated Design of Structural Frames", Quarterly Journal of Mechanics and Applied Mathematics, Vol. 9, 1956, pp. 257-278.

Miu84 Miura, H., "Overview: Applications of Numerical Optimization Methods to Helicopter Design Problems", Recent Experiences in Multidisciplinary Analysis and Optimization, NASA CP-2327, 1984.

Mor82 Morris, A. J., ed., Foundations of Structural Optimization: A Unified Approach, John Wiley and Sons, New York, N.Y., 1982.

Pee79 Peele, E.L., and Adams, W.M., Jr., A Digital Program for calculating the Interaction between Flexible Structures, Unsteady Aerodynamics and Active Controls, NASA TM-80040, January 1979.

Per49 Perkins, C.D. and Hage, R.E., Airplane Performance: Stability and Control (John Wiley and Sons, Inc., 1949, New York, N.Y.).

Per84 Perley, R., "Regression Analysis as a Design Optimization Tool", Recent Experiences in Multidisciplinary Analysis and Optimization, NASA CP-2327, 1984.

Pra56 Prager, W., "Minimum Weight Design of a Portal Frame", Proceedings of ASCE, Vol. 82, (EM4), 1956.

Pra84 Prasad, B. and Magee, C. L., "Application of Optimization Techniques to Vehicle Design - A Review", Recent Experiences in Multidisciplinary Analysis and Optimization, NASA CP-2327, 1984.

Pre86 Press, W.H., et. al., Numerical Recipes: The Art of Scientific Computing (Cambridge University Press, 1986, Cambridge, Mass.).

Rog81 Rogers, J. L., Dovi, A. R., and Riley, K. M., "Distributing Structural Optimization Software Between a Mainframe and a Minicomputer", Proceedings of the Engineering Software Second International Conference and Exhibition, London, England, March 1981.

Sch60 Schmit, L. A., "Structural Design by Systematic Synthesis", Proceedings, 2nd Conference on Electronic Computation, ASCE, New York, 1960.

Sch65 Schmit, L. A. and Thornton, W. Q., "Synthesis of an Airfoil at Supersonic Mach Number", NASA CR-144, Jan. 1965.

Sch74 Schmit, L. A. and Farshi, B., "Some Approximation Concepts for Structural Synthesis", AIAA Journal, Vol. 12, May 1974, pp. 692-699.

Sch76a Schmit, L. A. Jr., Miura, H., "Approximation Concepts for Efficient Structural Synthesis", NASA CR-2552, March 1976.

Sch76b Schmit, L. A. and Miura, H., "A New Structural Analysis/Synthesis Capability - ACCESS 1", AIAA Journal, Vol. 14, May 1976, pp. 661-671.

Sch81 Schmit, L. A., "Structural Synthesis - Its Genesis and Development", AIAA Journal, Vol.19, No.10 Oct. 1981, pp. 1249-1263.

Sen88 Sensburg, O., Fulhas, K., and Schimdinger, G., "Interdisciplinary Design of Aircraft Structures for Minimum Weight", AIAA Paper No. 88-2302, April, 1988.

Sha52 Shanley, F. R., Weight Strength Analysis of Aircraft Structures (McGraw Hill Book Company., Inc., 1952, New York, N.Y.).

Sme84 Smetana, F.O., Computer Assisted Analysis of Aircraft Performance Stability and Control (McGraw-Hill, Inc., 1984, New York, N.Y.).

Sob82a Sobieszczanski-Sobieski, J., "A Linear Decomposition Method for Large Optimization Problems - Blueprint for Development", NASA TM-83248, 1982.

Sob82b Sobieszczanski-Sobieski, J., Barthelemy, J.-F., and Riley, K. M., "Sensitivity of Optimum Solutions of Problem Parameters", AIAA Journal, Vol. 20, Sept. 1982, pp. 1291-1299

Sob88a Sobieszczanski-Sobieski, J., "Optimization by Decomposition: A Step from Hierarchic to Non-Hierarchic Systems", Recent Advances in Multidisciplinary Analysis and Optimization, NASA CP-3031, 1988, Part 1.

Sob88b Sobieszczanski-Sobieski, J., Bloebaum, C. L., Hajela, P., "Sensitivity of Control-Augmented Structure Obtained by a System Decomposition Method", AIAA Paper Number 88-2205, 1988.

Sob89 Sobieszczanski-Sobieski, J., "Multidisciplinary Optimization for Engineering Systems: Achievements and Potentials", NASA TM-101566, 1989.

Sob90 Sobieszczanski-Sobieski, J., "On the Sensitivity of Complex, Internally Coupled Systems", AIAA Journal, Vol. 28, Jan. 1990, pp. 153-160.

Sta79 Starnes, J.H. Jr., Haftka, R.T., Preliminary Design of Composite Wings for Buckling, Stress, and Displacement Constraints, Journal of Aircraft, Vol. 16, Aug. 1979, pp. 564-570.

Sto74 Storaasli, O.O., Sobieszczanski-Sobieski, J., On the Accuracy of the Taylor Approximation for Structure Resizing, AIAA Journal, Vol. 12, Feb. 1974, pp. 231-233.

Str69 Stroud, W. J. and Sykes, N. P., "Minimum Weight Stiffened Shells with Slight Meridional Curvature Designed to Support Axial Compressive Loads", AIAA Journal, Vol. 7, Aug. 1969, pp. 1599-1601.

Tol85 Tolson, R. H. and Sobieszczanski-Sobieski, J., "Multidisciplinary Analysis and Synthesis: Needs and Opportunities", AIAA Paper No. 85-0584, 1985.

Ton87 Tong, C., "Toward an Engineering Science of Knowledge-Based Design", Artificial Intelligence in Engineering, Vol. 2, Mar. 1987, pp. 133-166.

Van73 Vanderplaats, G., "CONMIN - A FORTRAN Program for Constrained Function Minimization – User's Manual", NASA TM X-62282, 1973.

Van84 Vanderplaats, G. N., Numerical Optimization Techniques for Engineering Design: With Applications (McGraw-Hill Book Company, 1984, New York, N.Y.).

Wei86 Weisshaar, T. A., Newsom, J. R., Zeiler, T. A., and Gilbert, M. G., "Integrated Structure/Control Design - Present Methodology and Future Opportunities", ICAS Paper No. 86-4.8.1, 1986.

Whe83 Whetstone, W. D., EISI - EAL Engineering Analysis Language Reference Manual -- EISI - EAL System Level 2091, Engineering Information Systems, Inc., July, 1983.

Wil78 Wilde, D. J., Globally Optimal Design (John Wiley and Sons, 1978, New York, N.Y.).

★U.S. GOVERNMENT PRINTING OFFICE: 1991 -627 -064/46009

Report Documentation Page

NASA National Aeronautics and Space Administration

1. Report No.	2. Government Accession No.	3. Recipient's Catalog No.
NASA CR-4413		

4. Title and Subtitle

Formal and Heuristic System Decomposition Methods in Multidisciplinary Synthesis

5. Report Date

December 1991

6. Performing Organization Code

7. Author(s)

Christina L. Bloebaum

8. Performing Organization Report No.

10. Work Unit No.

506-43-41

9. Performing Organization Name and Address

University of Florida
Dept. of Aerospace Eng., Mechanics & Eng. Science
Gainesville, FL 32611

11. Contract or Grant No.

NAG1-1004

13. Type of Report and Period Covered

Contractor Report

12. Sponsoring Agency Name and Address

NASA Langley Research Center
Hampton, VA 23665

14. Sponsoring Agency Code

15. Supplementary Notes

Technical Monitor - Dr. Jaroslaw Sobieski, Langley Research Center
A Dissertation presented to the graduate school of the University of Florida in partial fulfillment of the requirements for the Degree of Doctor of Philosophy, 1991.

16. Abstract

The multidisciplinary interactions which exist in large scale engineering design problems provide a unique set of difficulties. These difficulties are associated primarily with unwieldy numbers of design variables and constraints, and with the interdependencies of the discipline analysis modules. Such obstacles require design techniques which account for the inherent disciplinary couplings in the analyses and optimizations. The objective of this work was to develop an efficient holistic design synthesis methodology that takes advantage of the synergistic nature of integrated design.

A general decomposition approach for optimization of large engineering systems is presented. The method is particularly applicable for multidisciplinary design problems which are characterized by closely coupled interactions among discipline analyses. The advantage of subsystem modularity allows for implementation of specialized methods for analysis and optimization, computational efficiency, and the ability to incorporate human intervention and decision making in the form of an expert systems capability. The resulting approach is not a method applicable to only a specific situation, but rather, a methodology which can be used for a large class of engineering design problems in which the system is non-hierarchic in nature.

17. Key Words (Suggested by Author(s))	18. Distribution Statement
Optimization Decomposition Multidisciplinary optimization Design Synthesis	Unclassified - Unlimited Subject Category 05

19. Security Classif. (of this report)	20. Security Classif. (of this page)	21. No. of pages	22. Price
Unclassified	Unclassified	164	A08

NASA FORM 1626 OCT 86

For sale by the National Technical Information Service, Springfield, Virginia 22161-2171