SIEMENS

Design Sensitivity and Optimization User's Guide

Contents

Proprietary & Restricted Rights Notice

Chapter 1: Getting Started

1.1 Introduction to Design Sensitivity and Optimization

This chapter introduces some of the basic concepts of numerical optimization with an emphasis on the methods used in NX Nastran.

Some of the questions answered in this chapter include:

- What is design optimization, and how does it differ from analysis?
- What is the relationship between design sensitivity and optimization?
- How is an optimization problem formulated?
- How does an optimizer search for an optimum?
- How does an optimizer communicate with the structural analysis?

If you are interested in seeing some complete example problems in connection with the material covered in this chapter, you may want to refer to the first couple of examples from Example Problems. You will probably also need to refer to the Bulk Data descriptions in the *NX Nastran Quick Reference Guide* for the details of the entries used. Example Problems should help to give you some idea of what NX Nastran design optimization input and output looks like. The details are covered in later chapters.

What is Design Sensitivity and Optimization?

Design sensitivity and optimization are two separate, though closely related, topics.

For a given design, a design sensitivity analysis computes the rates of change of structural responses with respect to changes in design parameters. These design parameters are usually referred to as design variables and can be used to represent shell thicknesses, beam cross sectional dimensions, journal bearing sizes, and so on. In civil engineering, we may be interested in how changes in the deflection of a bridge span can be affected by changes in the dimensions of the bridge sections. In automotive design, we may want to investigate changes in cabin resonant frequencies given changes in panel thicknesses. These rates of change (what we call “partial derivatives” in the language of calculus) are called design sensitivity coefficients.

Design optimization refers to the process of generating improved designs. In NX Nastran, design optimization is performed by an optimizer. An optimizer is really nothing more than a formal plan, or algorithm, that is used to search for a “best” design. Design sensitivity coefficients are used in NX Nastran to assist the optimizer in this search process. Once these rates of change are known, the optimizer can, for example, find the optimal set of panel thicknesses that yield the lowest level of cabin resonant frequencies.

Why Use Design Sensitivity and Optimization?

Design sensitivity and optimization are used when we seek to modify a design whose level of structural complexity exceeds our ability to make appropriate design changes. What is surprising is that an extremely simple design task may easily surpass our decision-making abilities. Experienced designers, those with perhaps decades of experience, are sometimes fantastically adept at poring through mounds of data and coming up with improved designs. Most of us, however, cannot draw upon such intuition and experience. A basic goal of design optimization is to automate the design process by using a rational, mathematical approach to yield improved designs. Ways in which this might be put to use include:

1. Producing more efficient designs having maximum margins of safety.
2. Performing trade-off or feasibility studies.
3. Assisting in design sensitivity studies.
4. Correlating test data and analysis results (model matching).

In addition to providing a complete description of the optimization tools in NX Nastran, part of the aim of this user’s guide is to suggest various ways in which design sensitivity and optimization might be used. Consider the following examples:

Example A

A complex spacecraft is in a conceptual design stage. The total weight of the spacecraft cannot exceed 3,000 pounds. The nonstructural equipment including the payload is 2,000 pounds. Static loads are prescribed based on the maximum acceleration at launch. Also, the guidance systems require that the fundamental elastic frequency must be above 12 Hz. It is extremely important to reduce the structural weight since it costs several thousand dollars to place one pound of mass in a low earth orbit. There are three types of proposed designs: truss, frame, and stiffened shell configurations. Currently all of the designs fail to satisfy at least one design requirement and are expected to be overweight. We are to determine which configuration(s) promises the best performance and warrants detailed design study. Also, the payload manager needs to know how much weight could be saved if the frequency requirement were to be relaxed from 12 Hz to 10 Hz. The spacecraft’s structure contains about 150 structural parameters, which we may want to vary simultaneously.

Example B

One part of a vehicle’s frame structure was found to be overstressed. Unfortunately, it is too expensive to redesign that particular frame component at this stage in the engineering cycle. However, other structural components nearby can be modified without severe cost increases. There are nearly 100 structural design parameters that can be manipulated. The design goal is to reduce the magnitude of the stresses by reducing the internal load to the overstressed member.

Example C

A frame structure, which supports a set of sensitive instruments, must withstand severe in-service dynamic loads. Modal test results are available from comprehensive tests performed on the prototype structure. We need to create a finite element model for dynamic analysis that is much less detailed than the original model created for stress analysis since the costs of dynamic analysis using a complex model would be prohibitive. We must ensure, however, that the first ten modes obtained from our simplified model are in close agreement with those obtained from the test results. The goal is to determine suitable properties for the lumped quantities in our simplified dynamic model so that the first ten eigenvalues correlate well with the prototype.

How Does Design Optimization Differ from Analysis?

Although design optimization and analysis can be viewed as complementary, there are some important conceptual differences between the two that must be clear in order to make effective use of both.

Analysis Models

When we perform an "analysis," we create a mathematical idealization of some physical system in order to obtain estimates of certain response quantities. The class of responses that we are interested in defines the applicable analysis discipline to be used, while the accuracy of these responses is dependent on the quality of the analysis model and our general knowledge of the true system. Our choice of finite element types, representation of boundary conditions, loads, and definition of the finite element mesh all play critical roles in determining how well our model is able to predict the responses of the physical structure. The goal is to obtain an accurate prediction of the responses that can be expected from the real structure. For example, consider the plate subjected to uniform tensile loads in Figure 1-1. The corresponding analysis model in Figure 1-2 is a discretized finite element representation of idealized geometry, loads, and boundary conditions.

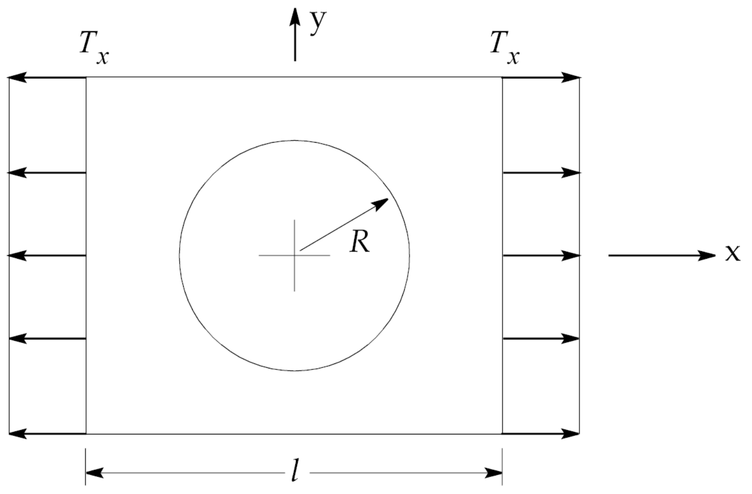
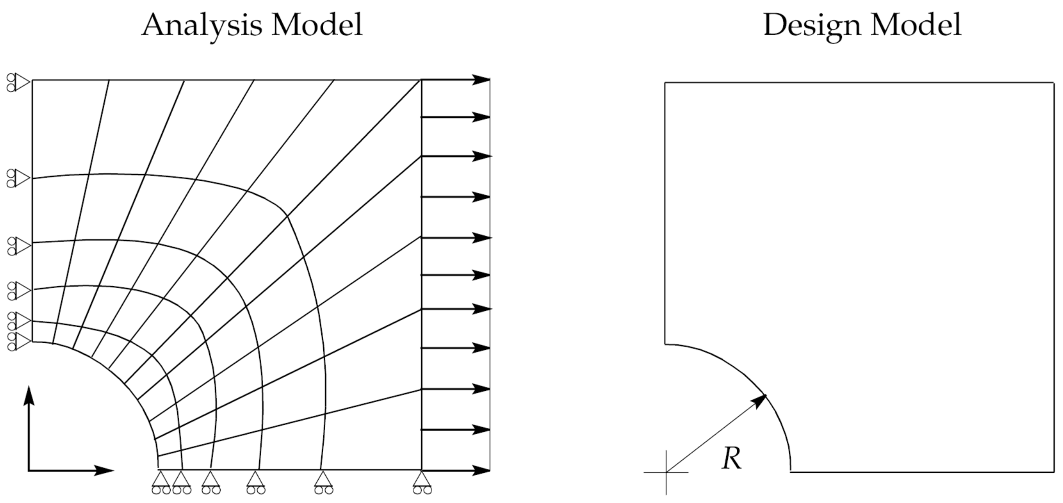


Figure 1-1. Flat Plate with Hole



Finite element discretization of:

- Structure (Mesh).
- Loads.
- Boundary Conditions (1/4 Plate Representation).

Find R such that:

- Weight is minimized.
- Stresses do not exceed allowables.

(R is the design variable, weight is the design objective, and stresses are the design constraints.)

Figure 1-2. Analysis Versus Design Models

Design Models

In contrast, a design model is an idealized statement of changes that might be made to the structure to improve its performance or response. In order to accomplish this, we need to define what we mean by an improved design. It may be the minimum weight or maximum stiffness, but whatever our choice is, this constitutes a statement of the design objective. The design may be varied such that certain bounds on responses are not exceeded. Expressions of maximum allowable stresses or minimum permissible frequencies are termed design constraints. And, in our description of how the design might be changed, we use design variables to express what we mean by a suitable variation. By convention, the mathematical region over which our design variables, objective, and constraints are defined is called the design space. Figure 1-2 also shows the design model corresponding to a redesign of the hole radius for weight minimization. It gives allowable structural variations (hole radius) subject to limits on structural responses (stress).

Probably the biggest difference between analysis and design is that analysis leads to “the solution” (within the limits of the analysis model), while design optimization leads to “a solution.” In other words, in analysis, we are usually guaranteed a unique solution, while more than one solution may be possible in design optimization. Mathematically, we can say that our design space may contain relative minima. This is analogous to the situation in which we may want to find the low point in a valley, yet various low points are separated by hills that we cannot see over. Simply finding a low point itself represents an acceptable solution, but there may be more than one such solution. In some instances we may be able to restate the problem and, in effect, shift the locations and contours of the hills to allow more efficient convergence. However, the fact remains that our goal has been stated in the context of design improvement and not in determining a unique solution (as in the case of analysis).

Integrity of Analysis and Design Modes

One thing the analysis and design models share is that the results are dependent upon the skill and judgement used in their construction. A poorly-meshed finite element model may lead to inaccurate and misleading results. Similarly, an ill-posed design optimization task may produce unexpected or useless results. Since the redesign process is based on analysis results, the results of design optimization are strongly dependent on the integrity of the analysis model.

Optimizer Limitations

A numerical optimizer seeks to find an improved design by trying to minimize or maximize a prespecified objective. Throughout this process, it must adhere to the bounds on responses and design variables given in the design model. It does not have the intelligence to modify the objective or relax any of these limits. For example, suppose you asked a friend to find you a nice apartment on his street. Your friend, the optimizer, may have a somewhat different definition of “nice” than you do. His income might be higher than yours, so that the optimal design he proposes may be infeasible in terms of your bank account. Even though he is searching just on his street, the next block may turn out to have an apartment that you consider a better value. The optimizer is not able to go beyond your specifications to search out other possible configurations.

The optimization problem statement requires an explicit description of the design objective, as well as bounds that define the region in which it may search. You may ask for a design satisfying a minimum flexibility requirement such as wing tip deflection, but without a weight budget, the design that the optimizer proposes may turn out to be unrealistic. You might get an extremely stiff, 200 ton wing out of this process. You also might have asked for a minimum weight design but allowed for negative physical dimensions. The optimizer may have no trouble minimizing the weight by adding negative mass, but this design may produce a true engineering challenge on the factory floor.

An optimizer is only able to search for your definition of a best design within the region you have defined. Configuration or trade-off studies cannot be directly addressed by a numerical optimizer, although an optimizer can be used to investigate the benefits of one design over another. An optimizer cannot weigh the benefits of a cast aluminum versus a welded steel support member, but it can tell you the best that each design is capable of and then let you decide which design path to pursue. Some of these considerations are, in fact, active areas of research. Even with these limitations, design optimization offers an extremely powerful set of design tools.

What Do I Need to Use Design Optimization Effectively?

Analysis Skills

Since design sensitivity and optimization depends on the results of analysis, a reasonable level of skill preparing NX Nastran analysis models is required. Design optimization can use analysis results from statics, normal modes, buckling, direct and modal frequency, modal transient, acoustic, and aeroelastic response analysis. In addition, design models can also employ superelements. Proficiency in any or all of these disciplines is useful.

Material Presented in This Guide, and Other Resources

The topics covered in this user's guide are intended to describe design sensitivity and optimization in NX Nastran. No prior knowledge of structural optimization is required. This guide attempts to describe all aspects completely so that someone new to the field of design optimization, as well as those with more experience, have enough information available to use it wisely and effectively.

Engineering Judgement and Common Sense

Probably the most critical requirement in the effective use of design optimization is common sense. Any sufficiently general and powerful tool possesses the capacity for misuse. This is especially true of numerical optimizers. The old adage that engineering is more art than science is probably more true of design optimization than of many other disciplines. As you gain experience with numerical optimization, you will probably discover a few pitfalls that are particular to your fields of application, and in the process you may discover some especially useful and efficient "tricks" for clearly stating your optimization tasks. In any field of application, there is no substitute for a well-posed problem. If your design goals are clear, the constraints are meaningful and well-conditioned, and the design variables are chosen carefully to produce useful designs, then success becomes more certain. The description of methods for accomplishing these goals forms the majority of the material of this guide. Before moving on to a detailed discussion of NX Nastran design optimization in Design Modeling for Sensitivity and Optimization and Design Sensitivity and Optimization in NX Nastran, the next two sections introduce some of the basic concepts of numerical optimizers and how they are used to solve problems in structural optimization.

1.2 Numerical Optimization Basics

This section introduces some of the basics of numerical optimization in an intuitive manner, stressing overall concepts over technical details. This section may be a useful introduction to numerical optimization for those entirely new to the subject. Once the basics of numerical optimization have been covered, Structural Optimization introduces the extension of these methods to the field of structural design. These sections in combination should provide adequate preparation for Design Modeling for Sensitivity and Optimization, which describes design modeling in NX Nastran.

How Much Do I Need to Know About Optimizers?

The safest answer to this question is, "The more you know about numerical optimizers, the better off you are." This answer does not imply that you must be an expert in numerical optimization techniques in order to use those implemented in NX Nastran. In fact, it would be nice if design optimization were such an automatic procedure that all an engineer had to do was to push the "optimization button" and an optimal design would result.

Some knowledge of the basic procedures involved in numerical optimizers will aid in understanding why an optimizer does what it does, in interpreting the final results of an optimization run, and in understanding what may have happened if the results are unexpected. For example, it is not necessary to know all of the details of sparse matrix decomposition in order to perform a linear static finite element analysis, but if singularities are present in your model and the decomposition procedure fails, the results are far less mystifying, and a modeling solution to the problem may be much more apparent if you know something of the basics of the solution procedure. Most of the parameters that control the optimizer in NX Nastran can be changed to improve performance for various classes of problems. Understanding the significance of these choices allows you to take full advantage of the tools at your disposal.

Design Optimization and Operations Research

Design optimization in structural redesign is actually an application of operations research (a branch of applied mathematics) to problems in engineering design. Generally, these are classes of problems in which the optimum allocation of scarce resources is desired. Operations research is frequently used to solve scheduling problems such as the routing of airplanes among various airport facilities. The allocation of 100 airplanes among 68 airports may not seem to have much to do with engineering design, but the optimization methods employed are similar and can be extended to structural problems. An efficient engineering solution to a design problem involves the optimum allocation of scarce resources. For example, tensile stresses cannot be allowed to assume unlimited values and must be restricted to within reasonable limits (the distribution of strain energy density must be made in an optimal manner). Likewise, reducing structural mass leads to a savings of material and possibly maintenance, fuel, or other indirect costs.

The Basic Optimization Problem Statement

Before introducing any formal relations, assume that we are assigned the type of task that a numerical optimizer might be asked to solve. We must examine ways to approach the problem. From this experience, we can build the equations that describe the basic optimization problem statement.

Objective and Constraint Functions, Design Spaces

Suppose we are standing on the side of a hill and would like to find the point of lowest elevation; this is our “objective.” Suppose also that some fences exist that force us to restrict our search to within the region enclosed by these fences. These fences or “constraints” act as bounds in our “design space,” which is the region that defines all of our possible positions on the hill. Only one out of all points on the hill can be considered an optimum, though. For simplicity, we are neglecting the presence of relative minima.

Finding the lowest point on the hill while staying inside the fences is no real problem. All we really need to do is have a good look about and note, from our perspective, which point on the hill appears to be the lowest. We have scanned the hill, analyzed thousands of possible candidates at a glance, and made an immediate decision. If we were blindfolded, though, our decision-making process would not be as simple, and that is exactly the task a numerical optimizer is faced with.

In a computational sense, the elevation of a single point on the hill, or the numerical value of our objective function, must be determined by an analysis that may take considerable effort. Evaluating hundreds or thousands of candidate designs may be prohibitive. We need a systematic method of searching for an optimal design. There are numerous techniques available to solve such a problem, all of which are classified as numerical optimization algorithms.

Search Directions Based on Gradient Information

Generally, numerical optimization methods seek to determine a direction of travel or “search direction” that moves us down the hill as quickly as possible, yet allows us to find an optimum that lies within the fences. A sequence of search directions is usually employed during the overall search procedure.

In our hill example, we could easily find a search direction, even though blindfolded, by taking small steps from side to side and then forward and back to test for elevation changes. Based on this estimate of a downhill direction, we can then proceed until we hit a fence or the hill starts to climb up again. What we have done is to find the local value of the "gradient" of our objective function and then used this information to establish a probable direction in which to search for a minimum. Numerical optimization algorithms that rely on gradient information are termed "gradient-based." Once we have done the best we can possibly do in this direction, we find another search direction, and again proceed as before. We continue to repeat this procedure until we cannot reduce the objective function any further.

Design Variables, Constrained, and Unconstrained Problems

To quantify the location of a point on the hill, we might use north-south and east-west coordinates corresponding to the elevation at a given point. In design optimization terms, this is a two "design variable" space since two coordinate values are required to uniquely specify a point in the design space. Two design variables are the most we can easily visualize. Considering that an optimizer may have to deal with tens or even hundreds of design variables, the task becomes understandably more complex. We might also have the condition that fences, or constraints, do not exist or are located so far "uphill" that they do not affect our search for a minimum. This situation is an unconstrained optimization task in contrast to a constrained task.

Equality Constraints

We might also have the condition that we want the design to lie on some prescribed path or curve drawn on the hillside. This is an equality constraint. Note that if there are as many equality constraints as design variables, a unique solution exists (as long as the equalities are linearly independent). This solution can be found using standard algebraic methods. A finite element analysis belongs to this category of problem. When the number and type of constraints do not enable a direct, unique solution, the job becomes complex and numerical optimizers must be used.

The Basic Optimization Problem Statement

We are now in a position to express our optimization task in a quantitative form. This mathematical expression of the design problem is called the basic optimization problem statement and can be written as follows:

minimize

$$F(\vec{x}) \text{ objective}$$

Equation 1-1.

subject to

$$g_j(\vec{x}) \le 0 \quad \mathrm{j} = 1, \ldots, n_g \text{ inequality constraints}$$

Equation 1-2.

$$h_k(\vec{x}) = 0 \quad \mathrm{k} = 1, \ldots, n_h \text{ equality constraints}$$

Equation 1-3.

$$x_i^I \le x_i \le x_i^u \quad \mathrm{i} = 1, \ldots, n \text{ side constraints}$$

Equation 1-4.

where:

$$\vec{x} = \{x_1, x_2, \ldots, x_n\} \text{ design variables}$$

Equation 1-5.

Function Minimization Maximization

The objective function is the scalar quantity to be minimized. It is a function of the set of design variables. (Although we stated the problem as a minimization task, we can easily maximize a function by minimizing its negative.) Side constraints are placed on the design variables to limit the region of search, for example, to plate thicknesses that are nonnegative or tubes whose wall thicknesses are less than one-tenth of the outer radii. The inequality constraints represent the fences on the hill and are expressed in a less than or equal to zero form by convention. We have satisfied a constraint and are thus within the boundary defined by the fence if the constraint's value is negative. We have violated the constraint if its value is positive. The location of the *j*-th "fence" lies at $g_j(\vec{x}) = 0$. Equality constraints, if present, must be satisfied exactly at the optimal design.

Linear Versus Nonlinear Problems

The objective and constraint functions may either be linear or nonlinear functions of the design variables. If all of these functions are linear, we may use linear techniques to find an optimal solution if one exists. If just one of these functions is nonlinear, then search algorithms that can deal with this nonlinearity must be used. NX Nastran includes capabilities for solving both linear and nonlinear optimization problems.

As seen throughout the remainder of this user's guide, the basic optimization problem statement is used directly in NX Nastran and influences the nomenclature adopted here. For example, the design objective is defined in the Case Control Section by the DESOBJ command, while the design variables and constraints are defined in the Bulk Data Section using the DESVAR and DCONSTR entries respectively. We will see how these are used in Design Modeling for Sensitivity and Optimization.

In conjunction with problems in structural optimization, we will discuss design spaces, objectives, constraints, and design variables. It is useful to take a look at a simple problem that is not explicitly related to structural optimization, just to become familiar with these concepts.

Example

Consider the following optimization problem:

minimize the objective:

$$F(\vec{x}) = x_1 + x_2$$

Equation 1-6.

subject to the constraints:

$$g_1(\vec{x}) = \frac{1}{x_1} + \frac{1}{x_2} - 2 \le 0$$

Equation 1-7.

$$x_1 \ge 0.1 \quad x_2 \ge 0.1$$

Equation 1-8.

Feasible Designs

The objective and constraint functions are dependent on two design variables x_1 and x_2. The objective $F(\vec{x})$ is linear in the design variables, while the constraint $g_1(\vec{x})$ is nonlinear. Figure 1-3 shows the two variable design space, where shading is used to denote regions in which the constraint or side constraints on the design variables are violated. If no constraints are violated, we say the current design is feasible (although it is probably not optimal). A design is infeasible if one or more of the constraints are violated. For example, the point (2,2) is a feasible design, whereas the point (0.05,3) violates not only the inequality constraint but also the lower bound constraint on x_1.

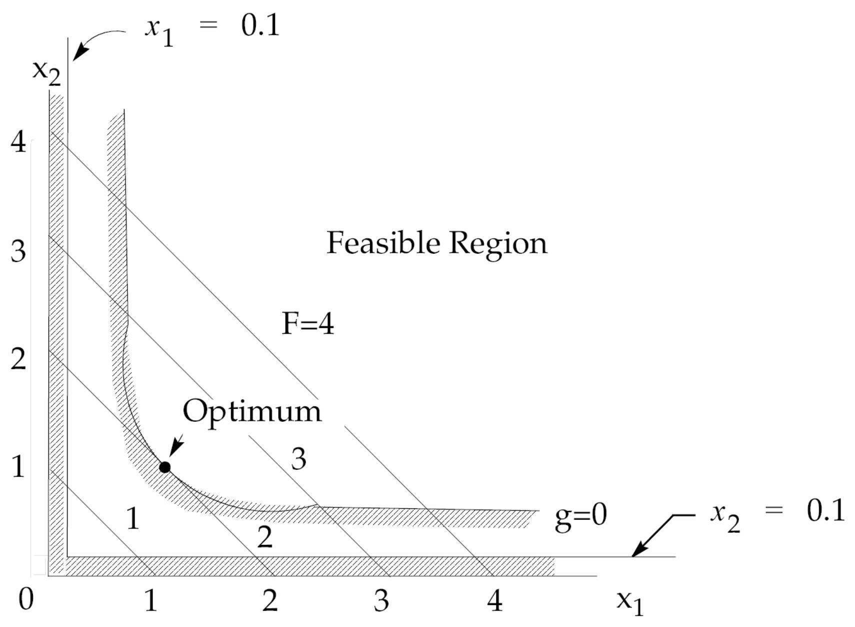


Figure 1-3. Two-Variable Function Space

The optimal design at (1,1) can be found by inspection. The explicit description of the design problem has allowed a graphical solution in two dimensions. In practice, we usually have more than two design variables and non-explicit constraints and objective function. This complexity requires an efficient searching procedure, recalling that the optimizer is essentially “blindfolded.”

Numerically Searching for an Optimum

Gradient-Based Algorithms

The optimization algorithms in NX Nastran belong to the family of methods generally referred to as “gradient-based,” since, in addition to function values, they use function gradients to assist in the numerical search for an optimum.

The numerical search process can be summarized as follows: for a given point in the design space, we determine the gradients of the objective function and constraints and use this information to determine a direction in which to search. We then proceed in this direction for as far as we can go, whereupon we investigate to see if we are at an optimum point. If we are not, we repeat the process until we can make no further improvement in our objective without violating any of the constraints.

Essentially this is the procedure used by the optimizer in NX Nastran, although the task is complicated by the structural optimization context. Here, we consider each of the aspects of this process in more detail, with an emphasis placed on the intuitive aspects rather than a rigorous mathematical treatment.

Finite Difference Gradient Approximations

The first step in a numerical search procedure is determining the direction to search. The situation may be somewhat complicated if the current design is infeasible (one or more violated constraints) or if one or more constraints are critical. For an infeasible design, we are outside of one of the fences, to use the hill analogy. For a critical design, we are standing right next to a fence. In general, we at least need to know the gradient of our objective function and perhaps some of the constraint functions as well. The process of taking small steps in each of the design variable directions (suppose we are not restricted by the fences for this step) corresponds exactly to the mathematical concept of a first-forward finite difference approximation of a derivative. For a single independent variable the first-forward difference is given by

$$\frac{df(x)}{dx} \cong \frac{f(x + \Delta x) - f(x)}{\Delta x}$$

Equation 1-9.

where the quantity Δ*x* represents the small step taken in the direction *x*. For most practical design tasks, we are usually concerned with a vector of design variables. The resultant vector of partial derivatives, or gradient, of the function can be written as

$$\nabla F(\vec{x}) = \begin{Bmatrix} \dfrac{\partial F}{\partial x_1} \\ \cdot \\ \cdot \\ \dfrac{\partial F}{\partial x_n} \end{Bmatrix} \cong \begin{Bmatrix} \dfrac{F(x + \Delta x_1) - F(\vec{x})}{\Delta x_1} \\ \cdot \\ \cdot \\ \dfrac{F(\vec{x} + \Delta x_n) - F(\vec{x})}{\Delta x_n} \end{Bmatrix}$$

Equation 1-10.

where each partial derivative is a single component of the dimensional vector.

Note

Details of how NX Nastran computes sensitivities can be found in Design Sensitivity Analysis.

Direction of Steepest Descent

Physically, the gradient vector points uphill, or in the direction of increasing objective function. If we want to minimize the objective function, we will actually move in a direction opposite to that of the gradient. The steepest descent algorithm searches in the direction defined by the negative of the objective function gradient, or

$$\vec{S} = -\nabla F$$

Equation 1-11.

since proceeding in this direction reduces the function value most rapidly. $\vec{S}$ is referred to as the search vector.

For now, just note that NX Nastran uses the steepest descent direction only when none of the constraints are critical or violated and then only as the starting point for other, more efficient search algorithms. The difficulty in practice stems from the fact that, although the direction of steepest descent is usually a very good starting point, subsequent search directions often fail to improve the objective function significantly. In NX Nastran we use other, more efficient methods that can be generalized for the cases of active and/or violated constraints. We will briefly introduce these methods later in this section. The next question to consider is: once we have determined a search direction, how can this be used to improve our design?

One-Dimensional Search

Just as in the hill example, once we found a search direction, we proceeded “downhill” until we bumped into a fence or until we reached the lowest point along our current path. Note that this requires us to take a number of steps in this given direction, which is equivalent to a number of function evaluations in numerical optimization. For a search direction $\vec{S}$ and a vector of design variables $\vec{x}$, the new design at the conclusion of our search in this direction can be written as

$$\vec{x}^1 = \vec{x}^0 + \alpha^* \vec{S}^1$$

Equation 1-12.

This relation allows us to update a potentially huge number of design variables by varying the single parameter α. We have been able to reduce the dimensionality from *n* to 1, that is, from *n* design variables to a single search parameter α. For this reason, this process is called a one-dimensional search. When we can no longer proceed in this search direction, we have the value of α, which represents the move required to reach the best design possible for this particular direction. This value is defined as α^*. The new objective and constraints can now be expressed as

$$F^1 = F(\vec{x}^0 + \alpha^* \vec{S}^1)$$

Equation 1-13.

$$g_j^1 = g_j(\vec{x}^0 + \alpha^* \vec{S}^1) \quad j = 1, \ldots, n_g$$

Equation 1-14.

From this new point in the design space, we can again compute the gradients and establish another search direction based on this information. Again, we will proceed in this new direction until no further improvement can be made, repeating the process, if necessary.

At some point we will not be able to establish a search direction that can yield an improved design. We may be at the bottom of the hill, or we may have proceeded as far as possible without crossing over a fence. In the numerical search algorithm, it is necessary to have some formal definition of an optimum. Any trial design can then be measured against this criteria to see if it is met and an optimum has been found. This required definition is provided by the Kuhn-Tucker conditions that are physically quite intuitive.

Convergence to an Optimum: The Kuhn-Tucker Conditions

Figure 1-4 shows a two design variable space with constraints $g_1(\vec{x})$ and $g_2(\vec{x})$ and objective function $F(\vec{x})$. The constraint boundaries are those curves for which the constraint values are identically zero. A few contours of constant objective are shown as well; these can be thought of as contour lines drawn along constant elevations of the hill. The optimum point in this example is the point that lies at the intersection of the two constraints. This location is shown as $\vec{x}^*$.

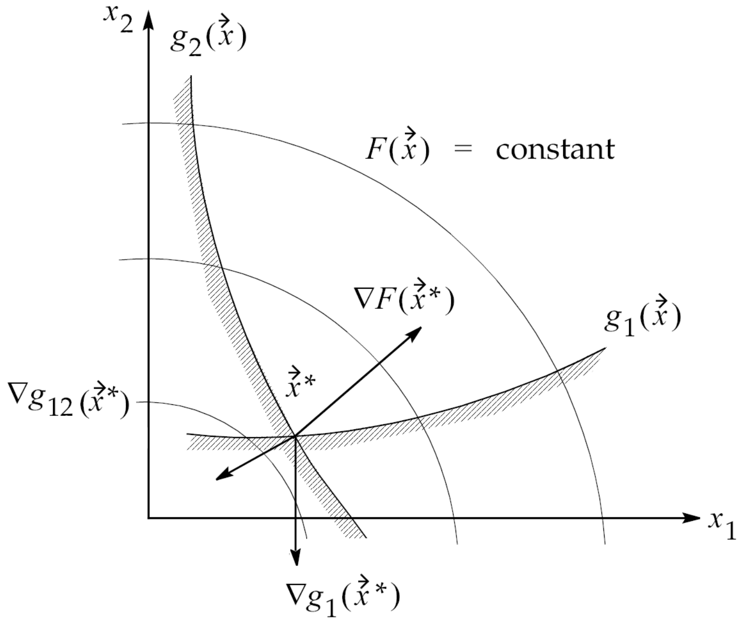


Figure 1-4. Kuhn-Tucker Condition at a Constrained Optimum

If we compute the gradients of the objective and the two active constraints at the optimum, we see that they all point off roughly in different directions. (Remember that function gradients point in the direction of *increasing* function values.) For this situation-a constrained optimum-the Kuhn-Tucker conditions state that the vector sum of the objective and all active constraints must be equal to zero given an appropriate choice of multiplying factors. These factors are called the Lagrange multipliers. (Constraints that are not active at the proposed optimum are not included in the vector summation.) Indeed, Figure 1-5 shows this to be the case where λ_1 and λ_2 are the values of the Lagrange multipliers that enable the zero vector sum condition to be met. We could probably convince ourselves that this condition could not be met for any other point in this design space.

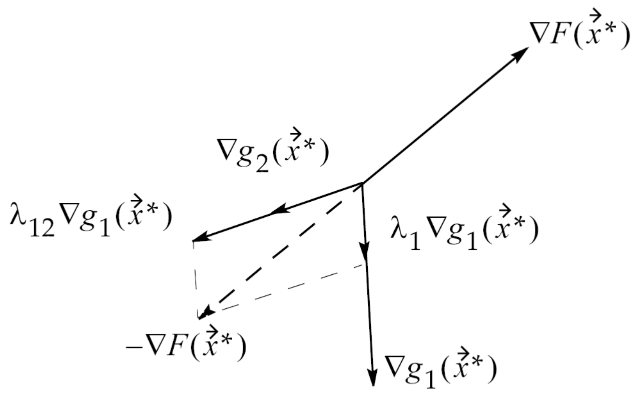


Figure 1-5. Graphical Interpretation of Kuhn-Tucker Conditions

The Kuhn-Tucker conditions are useful even if there are no active constraints at the optimum. In this case, only the objective function gradient is considered, and this is identically equal to zero; i.e., any finite move in any direction will not decrease the objective function. A zero objective function gradient indicates a stationary condition.

Not only are the Kuhn-Tucker conditions useful in determining if we have achieved an optimal design; they are also physically intuitive. The optimizer in NX Nastran tests the Kuhn-Tucker conditions in connection with the search direction determination algorithm. The interested reader can find the theoretical details in Appendix D.

A Simple Structural Example

In this section we covered the basic optimization problem statement, the concept of a design space, gradient-based search techniques, and the meaning of an optimum in terms of satisfying the Kuhn-Tucker conditions. We pause here to take a look at a simple example to help fit these pieces together and to introduce some qualitative aspects of the optimizer used in NX Nastran.

The cantilever beam in Figure 1-6 has a rectangular cross section. Suppose we want to minimize the volume, and thus the weight of the beam, subject to constraints on maximum bending stress and deflection due to the tip loading. In addition, the beam's cross section must remain below a maximum beam height-to-width ratio to guard against the introduction of twisting modes of failure.

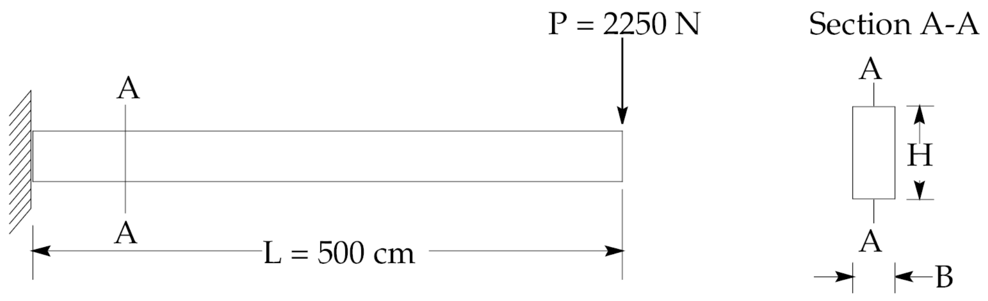


Figure 1-6. Cantilever Beam.

The optimization problem statement could be written as follows:

minimize

$$V = B \cdot H \cdot L$$

Equation 1-15.

subject

$$\sigma = \frac{Mc}{I} = \frac{6PL}{BH^2} \le 700$$

Equation 1-16.

$$\delta = \frac{PL^3}{3\mathrm{EI}} = \frac{4PL^3}{BH^3} \le 2.54$$

Equation 1-17.

$$\frac{H}{B} \le 12$$

Equation 1-18.

$$1 \le B \le 20$$

Equation 1-19.

$$20 \le H \le 50$$

Equation 1-20.

Since the objective and constraints are available explicitly, we can graphically display the two-variable design space as shown in Figure 1-7.

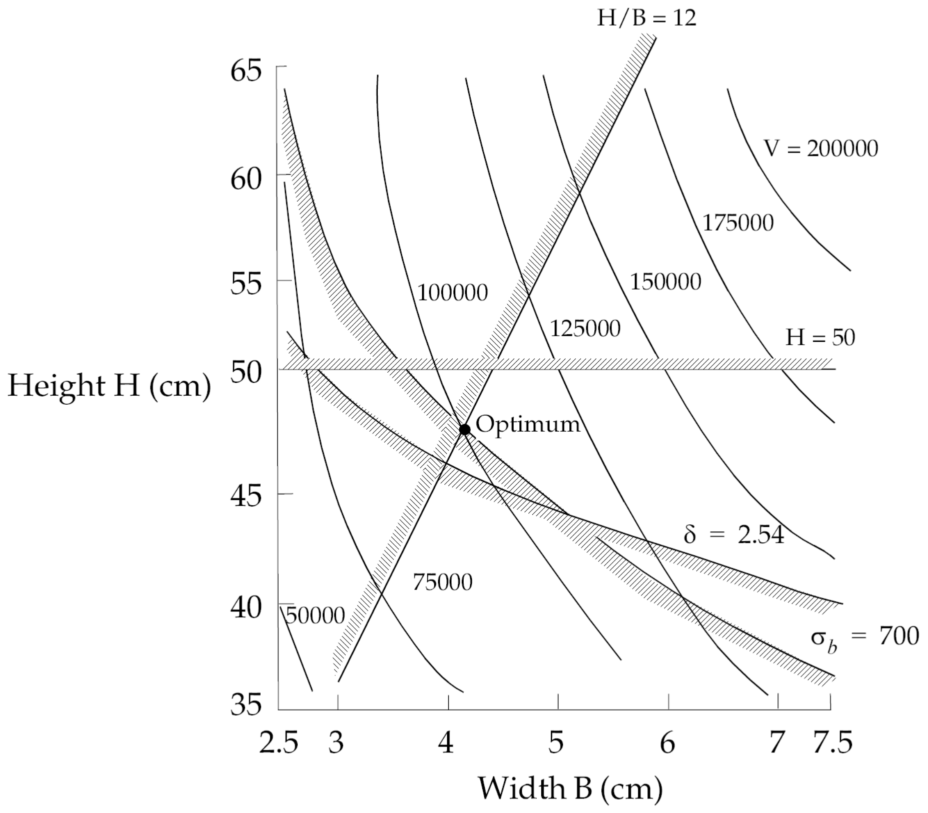


Figure 1-7. Cantilever Beam Design Space

In Figure 1-7, note that the optimum lies at the vertex formed by the intersections of the beam bending stress constraint and the constraint on maximum allowable ratio of cross-sectional height to width. This optimum occurs at an objective of approximately 1,000 cm^3. Let us examine the path that the optimizer might take as it searches for this constrained minimum.

Assume that we begin with an initial design of H = 44 cm and B = 7 cm. Since none of the constraints are active, the direction of steepest descent is used as an initial search direction, and the optimizer proceeds in this direction until it encounters a constraint boundary. From Figure 1-8 we see that the structural volume cannot be reduced any further in this search direction without violating the maximum allowable tip deflection requirement. The optimizer is now faced with a choice. A finite move in the direction of steepest descent would not be admissible since constraints would then be violated, yet we know that the objective function can still be reduced.

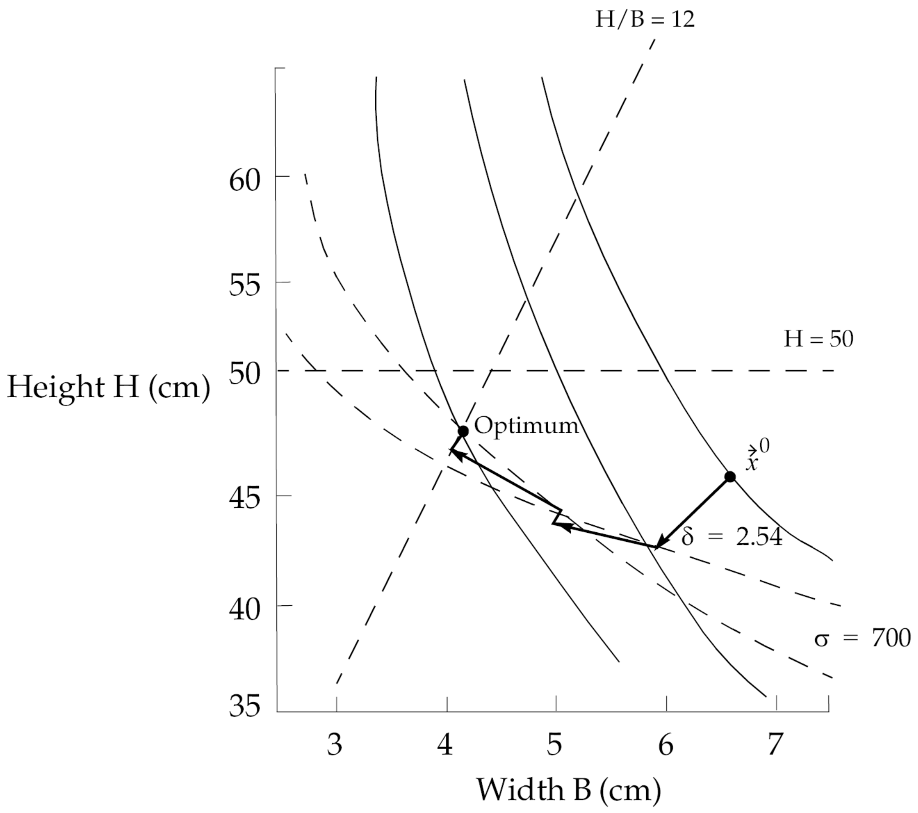


Figure 1-8. Sequence of Iterations: Modified Method of Feasible Directions.

The optimizer in NX Nastran resolves the situation by choosing a search vector that effectively follows the active constraint boundary in the direction of decreasing objective function. (If the optimizer could not find any direction in which to move, an optimum would be at hand since the Kuhn-Tucker conditions have implicitly been satisfied.) The objective can be reduced further and we observe that by the time two such iterations have been completed, the true optimum has been reached.

Default Optimizer in NX Nastran

Note that for both of these iterations, one or more constraints have been slightly violated in the interim. This is a characteristic of the default optimizer used in NX Nastran, the modified method of feasible directions, which establishes a search direction tangent to the critical constraint(s). If the constraint is nonlinear, a finite move in this direction may lead to a small constraint violation. Thus, continual corrections must be made by stepping back toward the constraint boundary along the current search direction. These corrections are performed as part of the search process, and are qualitatively represented in Figure 1-8 as small steps back to the constraint boundary.

Dealing with Initially Infeasible Designs

If the initial design is infeasible, the optimizer's first task is to return to the feasible region. Once this has been achieved, the optimizer can then proceed to minimize the objective, if possible. Often simply finding a design in which none of the constraints are violated is an engineering success. If no feasible designs are found, this still provides useful information about the original design formulation, as one or more of our performance criteria may need to be relaxed somewhat if we hope to produce a feasible design. By reexamining our design goals, we may learn something about the problem that was not evident before.

Cautionary Notes

A critical but related issue involves the identification of various applicable modes of failure. In developing a set of design criteria, we must ensure that all possible failure modes are adequately addressed by the design specifications. This is true in all aspects of engineering design but is especially so in design optimization. To use the current example, if we did not specify a maximum allowable beam height to width ratio, we might run the risk of introducing twisting or other buckling modes beyond the simple bending stress criteria we had already accounted for. Without this constraint, the optimizer would be able to reduce the structural volume even further (see the design space of Figure 1-7), but would be completely unaware of the introduction of other failure modes possible with narrow beam sections. Consequently, the engineer, not the optimizer, must accept ultimate responsibility for the integrity of the final design.

Summary

The intent of this example was to help illustrate the concepts presented in this section as well as to give a general idea of the approach used by the modified method of feasible directions. The discussion was simplified by the fact that we had an explicit functional description of the design space beforehand, as well as only two design variables. In any real structural optimization task, each of the data points in the design space can only be determined based on the results of a complete finite element analysis. This may be quite expensive. Also, since a numerical optimizer usually needs a number of these function evaluations throughout the search process, the costs associated with this analysis can quickly become enormous.

These factors combined with tens or even hundreds of design variables and thousands of constraints force us to consider methods for efficiently coupling structural analysis routines with numerical optimizers. The field of structural optimization is based on the introduction of approximation concepts, which reduce the need for repeated finite element function evaluations. Approximation concepts are actually quite intuitive and are the topic of the next section.

1.3 Structural Optimization

In Introduction and Numerical Optimization Basics, we introduced some examples of ways in which numerical optimization might be used to solve design problems and gave a brief overview of gradient-based numerical optimizers. These tools yield an orderly, rational approach to solving a minimization problem subject to a set of constraints; however, a large number of function evaluations may need to be performed. These function evaluations may be expensive, especially in a finite element structural analysis context. This section discusses methods used in NX Nastran to reduce these costs.

Basic Difficulties in Structural Optimization

Historically, the first attempts at linking structural analysis with numerical optimization were largely a direct coupling or “black box” type of approach as seen in Figure 1-9. Whenever the optimizer needed a function evaluation, the finite element analysis would be invoked to provide the necessary information. The sheer number of analyses quickly tended to make this approach useless in all but the smallest of problems. Recall that the numerical optimizer may not only request response derivatives with respect to the design variables, but a number of function evaluations must also be performed during each of the one-dimensional searches. This situation could quite easily lead to hundreds of analyses.

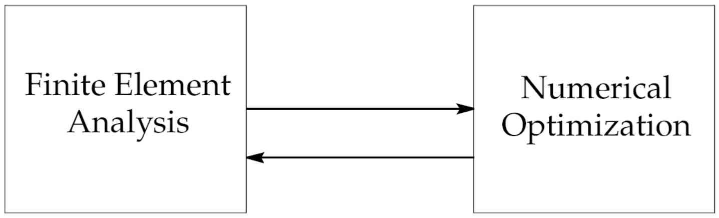


Figure 1-9. Early Structural Optimization Attempts

Structural Responses. Implicit Functions of the Design Variable

The principal complicating factor in structural optimization is that the response quantities of interest are usually implicit functions of the design variables. For example, a plate element's stress variation with changing thickness can only be determined in the general case by performing a finite element analysis of the structure. When we consider that the optimizer may be asked to deal with perhaps hundreds of design variables and thousands of constraints, it becomes apparent that we do not have the luxury of invoking a full finite element analysis each time the optimizer proposes an incremental design change.

To avoid these difficulties, certain approximations can be implemented to reduce the computational overhead. The remainder of Structural Optimization will introduce each of these methods, all of which are available in NX Nastran.

Overview of Approximation Concepts Used in Structural Optimization

Approximation concepts are actually nothing more than the computational implementation of methods generally used by experienced design engineers. In many instances, an engineer is handed a stack of analysis data and asked to propose an improved design. This raw data usually contains much more information than is necessary to suggest possible design improvements. The question becomes one of how to reduce the problem sufficiently so that only the most pertinent information is considered in the process of generating a better design.

Design Variable Linking

A first step may be to narrow the design task to that of determining the best combination of just a few design variables. There is virtually no way for a designer to consider fifty or one hundred variables simultaneously and expect to find a suitable combination from the group. It is much more efficient to link these together if possible. That is, it would be advantageous if all the design variables could be varied in a suitably proportional manner according to changes made to a much smaller set of independent variables. This way, a large number of structural properties might be varied according to a smaller set of well chosen variables. Describing a shape defining polynomial surface in terms of just a few characteristic parameters, or allowing only linear variations of plate element thicknesses, are both examples of types of design variable linking.

Constraint Deletion

The next step might then be to identify a few constraints that are violated or nearly violated. There may be just a few constraints that are currently guiding the design, such as stress concentrations due to thermal loads, while others, such as the first bending mode, may be nowhere near critical and can be temporarily disregarded. This is called a “constraint deletion” process. Constraint deletion allows the optimizer to consider a reduced set of constraints, simplifying the numerical optimization phase. This also reduces the computational effort associated with determining the required structural response derivatives (i.e., sensitivity analysis costs are reduced as well).

Formal Approximations

Once the engineer has determined the constraint set that seems to be driving the design, the next step might be to perform some sort of parametric analysis in order to determine how these constraints vary as the design is modified. The results of just a few analyses might be used to propose a design change based on a compromise among the various trial designs.

A parametric study of the problem is carried one step further in structural optimization with formal approximations, or series expansions of response quantities in terms of the design variables. Formal approximations allow us to construct an approximation to the true design space that, although only locally valid, is explicit in the design variables. The resultant explicit representation can then be used by the optimizer whenever function evaluations are required instead of the costly, implicit finite element structural analysis.

This coupling is illustrated in Figure 1-10. The finite element analysis forms the basis for creation of the approximate model that is subsequently used by the optimizer. The approximate model includes the effects of design variable linking, constraint deletion, and formal approximations. Design variable linking is established by the engineer, while constraint deletion and formal approximations are performed automatically in NX Nastran.

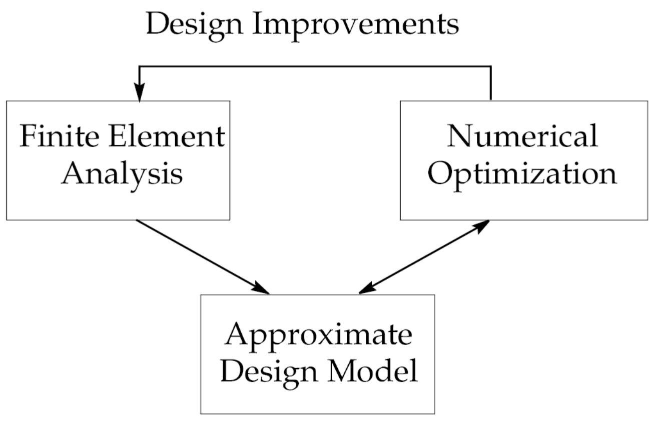


Figure 1-10. Coupling Analysis and Optimization Using Approximations

Design Cycles

Once a new design has been proposed by the optimizer (based on the information supplied by the approximate model) the next step would most likely be to perform a detailed analysis of the new configuration to see if it has managed to satisfy the various design constraints and reduce the objective function. This reanalysis update of the proposed designs using a complete finite element analysis is represented by the upper segment of the loop in Figure 1-10. If a subsequent approximate optimization is deemed necessary, the finite element analysis serves as the new baseline from which to construct another approximate subproblem. This cycle may be repeated as necessary until convergence is achieved. These loops are referred to as design cycles.

Design variable linking, constraint deletion, and formal approximations form the basis for the approximation concepts implemented in NX Nastran. The remainder of Structural Optimization takes a closer look at these methods.

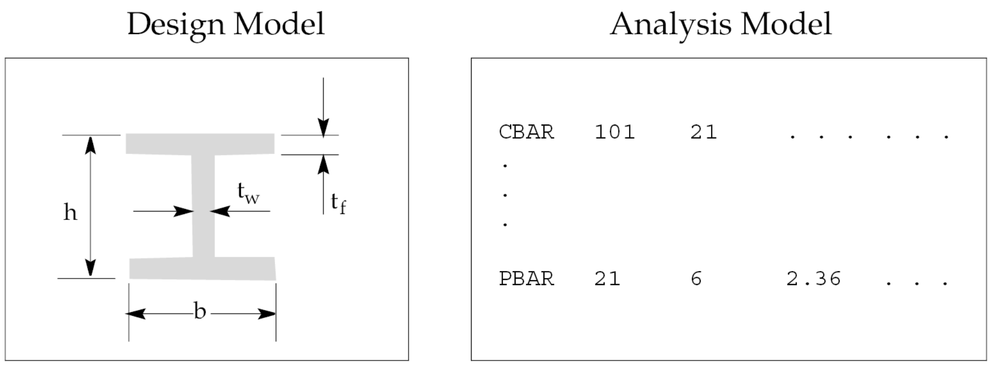
Design Variable Linking

Before discussing design variable linking, a few words should be said about how design modeling differs from analysis modeling.

Differences Between Design and Analysis Modeling

In analysis, our goal is to construct a mathematical idealization of an actual physical system. We must determine analysis model properties, select appropriate response quantities, and determine a suitable finite element mesh so that the desired responses are computed within an acceptable degree of accuracy. On the other hand, in design modeling we wish to define how our model can change in pursuit of an improved design along with the criteria that we (or the optimizer) use to judge the effects of these changes. Some formulations may be inherently more efficient than others.

For example, suppose we have a structure consisting of symmetric I-section beams as shown in Figure 1-11. For analysis purposes, we must specify the cross-sectional properties, A , I_1 , and I_2 on a property Bulk Data entry. However, from a design perspective, the actual cross-sectional dimensions are of greater interest. A design must ultimately be chosen from readily available manufactured I-beam sections and a list of optimal cross-sectional dimensions rather than properties will allow us to make the proper selection. The natural choice is to let the I-section dimensions be the design variables. In the design model we will specify the manner in which the I-section dimensions relate to the cross-sectional properties. The functional relationships are illustrated in Figure 1-11.



```
CBAR   101   21      . . .  . . .
.
.
.
PBAR   21    6       2.36    . . .
```

Design Variables — Analysis Model Parameters

$$h,\ b,\ t_w,\ t_f \longleftrightarrow A,\ I_1,\ I_2,\ I_{12},\ J,\ \ldots$$

$$A = 2bt_f + (h - 2t_f)t_w$$

$$I_1 = \frac{bh^3}{12} - \frac{(b - t_w)}{12}(h - 2t_f)^3$$

Figure 1-11. Design and Analysis Models for a Symmetric I-Beam Section

Note

These equations can be expressed in the design model using DVPREL2 and DEQATN Bulk Data entries. See Design Modeling for Sensitivity and Optimization.

Depending on the number of bar elements in our analysis model, we may not want to allow each to vary independently of the other in our design model. We will probably run into cost or manufacturing problems if we try to build a frame structure with 150 elements and 150 different cross sections. To impose some order on the design, it might be preferable to restrict the design to a set of, say, ten different sections. We can then arrange the elements so that all vary according to just this set of ten.

Linking by Property Entries

One way to link properties in NX Nastran is to use the analysis model property entries directly. In the analysis model, a property entry is defined and referenced by a number of elements. In the design model, design variables are related to the analysis model property entries. Thus, all elements that reference a particular property entry vary in unison as a function of the design model description. This is shown in the diagram of Figure 1-12, and the approach is discussed in Overview of Design Modeling.

Note

DVPREL1 and DVPREL2 entries are discussed in Design Modeling for Sensitivity and Optimization and Input Data.

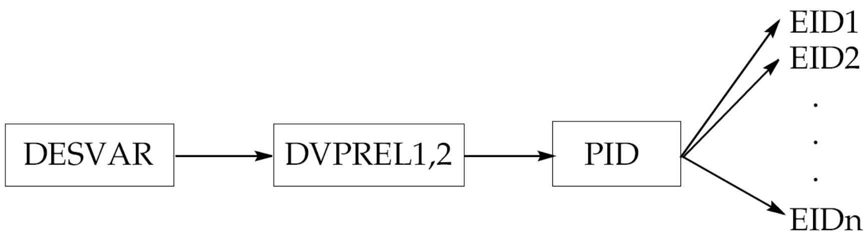


Figure 1-12. Property Entry Linking

Explicit Design Variable Linking

Another design variable linking technique is to express a design variable as a linear function of other design variables. This type of relation introduces the concept of an independent design variable set and a dependent design variable set, and can be written as

$$\{x\} = \left\{\begin{array}{c} x_I \\ \hline x_D \end{array}\right\} = \left\{\begin{array}{c} 0 \\ \hline C_O \end{array}\right\} + \left[\begin{array}{c} I \\ \hline C_D \end{array}\right]\{x_I\}$$

Equation 1-21.

where the dependent design variable set $\vec{x}_D$ is a linear function of the independent design variable set $\vec{x}_I$. These relations can be specified using DLINK (Design variable LINKing) Bulk Data entries. This specification reduces the order of the design space since only the design variables in the independent set are considered by the optimizer.

Apart from numerical efficiency, there may be other design-related reasons for introducing some type of design variable linking. In the design of the transmission tower in Figure 1-13, we may want to enforce sizing symmetry about the vertical axis. For shape optimization we may want to enforce symmetry about the y-axis in addition to the requirement that the uprights remain straight.

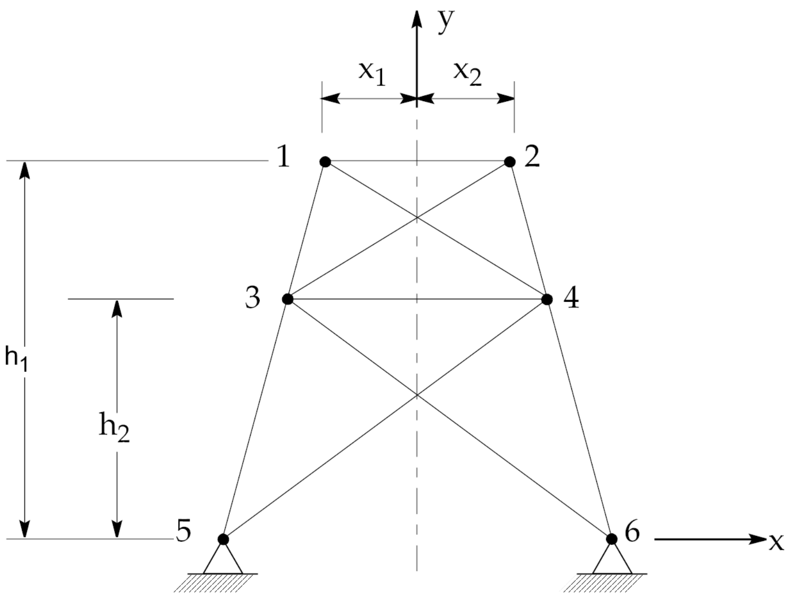


Figure 1-13. Symmetric Transmission Tower

Assume, for example, that the variables x_1 though x_6 are used to describe the *x*-component variations in the dimensions of the tower. We may want to link the x_i variables such that

$$\left.\begin{aligned} x_1 &= -x_2 \\ x_5 &= -x_6 \end{aligned}\right\} \text{symmetry}$$

$$\left.\begin{aligned} x_3 &= -x_4 \\ x_4 &= \frac{h_2}{h_1}x_2 + \left(1 - \frac{h_2}{h_1}\right)x_6 \end{aligned}\right\} \text{straight sides}$$

Equation 1-22.

From these relations, note that the independent and dependent variable sets are

$$\begin{aligned} \{x\}_I &= \begin{bmatrix} x_2 & x_6 \end{bmatrix}^T \\ \{x\}_D &= \begin{bmatrix} x_1 & x_3 & x_4 & x_5 \end{bmatrix}^T \end{aligned}$$

Equation 1-23.

Not only have our structural redesign conditions been met using these simple relations, but the order of the design space has also been reduced from six variables to two.

Constraint Regionalization and Deletion

If the number of constraints can be temporarily reduced, we would expect a reduced computational cost at both the numerical optimization level and, more importantly, at the sensitivity analysis level. This is because we reduce the number of structural responses for which gradient information must be computed.

Normalized Constraints in NX Nastran

In NX Nastran, constraints are internally generated in a normalized form, based on upper and lower response bounds selected by the engineer. Constraints may be present that depend on displacement, stress, eigenvalue responses, and so on. These response types may vary by orders of magnitude. To allow all constraints to be treated equally, regardless of the response magnitude, constraints in NX Nastran are normalized by the absolute values of the bounds. The result is that a constraint having a value of +1.0, for example, is violated by 100%, while a constraint with a value of -0.5 is only 50% of its critical value (0.0). Upon scanning the values of all the normalized constraints, we may find that we only need consider a subset above a particular threshold. Constraints below this value might then be temporarily disregarded on the assumption that they are not currently driving the design.

Note

This topic is discussed in Design Sensitivity and Optimization in NX Nastran in greater detail, but the general idea can be given here. Defining the Constraints discusses constraint generation in NX Nastran in greater detail.

Note

A negative value of a normalized constraint indicates constraint satisfaction, while a positive value indicates violation. See the Basic Optimization Problem Statement.

Figure 1-14 illustrates screening based on normalized constraints by representing the current value of each constraint function in a bar chart format. If any constraint exceeds the “truncation threshold” value denoted by TRS, we retain it for the ensuing approximate optimization, while temporarily deleting all others below TRS.

Constraint Screening

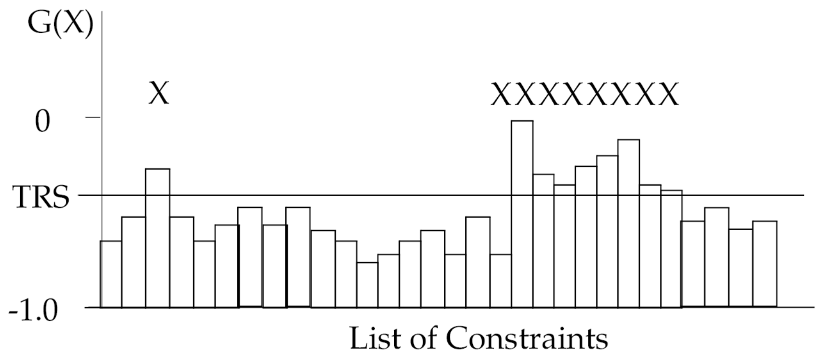


Figure 1-14. Constraint Deletion

Note

The default value for TRS is -0.5, but may be changed using the DSCREEN entry. See Approximation Concepts in Design Optimization for more details.

If there are a large number of constraints below this truncation threshold (as is typical in structural optimization), the constraint screening phase will greatly simplify the optimization task. However, the number of constraints can still be further reduced using regionalization.

Constraint Regionalization

Suppose we have a particular section of an airplane wing skin, modeled with a number of quadrilateral elements. This wing skin is to be of constant thickness for manufacturing considerations and is found to be overstressed in a number of locations. It would not make much sense to retain the stress constraints for every element in the skin panel since all of these stresses will vary nearly in unison as the panel thickness is varied. The constraints from all of the neighboring elements will likely contain redundant information. Thus, it is probably safe to retain only a few of the largest valued constraints from this “region.”

Constraint regionalization is shown in Figure 1-16, which is the same as Figure 1-15, but with the constraints grouped into three regions. For now, we will assume these regions have been established based on some design model characteristics. Of the three regions shown, only two have constraints that are numerically greater than the truncation threshold. These constraints will pass the first screening test. Since the retained constraints within each region are likely to contain redundant information, only the largest constraints from each region are retained. These constraints are denoted by the check marks in the figure. NSTR, which in this example is 2, stands for the maximum number of constraints to be retained per region.

Note

NSTR and TRS defaults can be overridden using the DSCREEN Bulk Data entry. The number of regions are implicitly defined in connection with the DRESP1 Bulk Data entry.

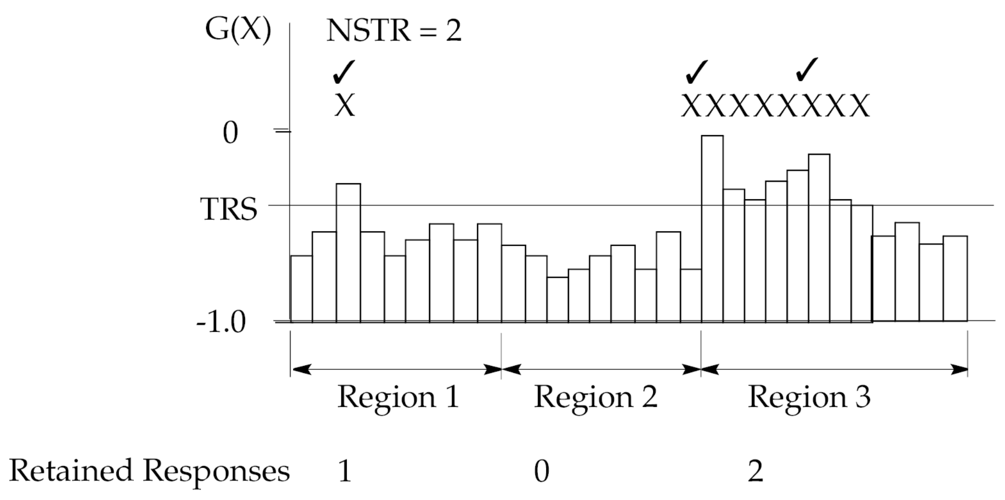


Figure 1-15. Constraint Regionalization

If NR is equal to the number of regions in our design model and NSTR is the maximum number of constraints to be retained per region, then the maximum number of retained constraints is NR · NSTR. This is often one or two orders of magnitude less than the number of constraints in the original set.

Formal Approximations

By applying constraint regionalization and deletion, we can reduce the number of constraints to only that small subset that is necessary to adequately guide the design. However, we would still like to replace the implicit and costly finite element analyses with explicit approximations for the objective and constraint functions.

The approximating functions used in NX Nastran are based on Taylor series expansions of the objective and constraints. For any function $f(x)$, an infinite series about a known value $f(x^0)$ in terms of the change in the independent variable Δx can be written as

$$(f(x^0 + \Delta x) = f(x^0)) + \left.\frac{df}{dx}\right|_{x^0} \cdot \Delta x + \left.\frac{d^2 f}{dx^2}\right|_{x^0} \cdot \frac{\Delta x^2}{2!} + \left.\frac{d^3 f}{dx^3}\right|_{x^0} \cdot \frac{\Delta x^3}{3!} + \ldots$$

Equation 1-24.

In addition to the function value $f(x^0)$, this series requires that all derivatives at x^0 be known as well. Determining these derivatives may present some difficulty, so the series is often truncated to a given power in Δx , yielding an approximate representation of the original function.

For example, if we only included through the first derivative term in the series, we would obtain a linear approximation:

$$\tilde{f}(x^0 + \Delta x) = f(x^0) + \left.\frac{df}{dx}\right|_{x^0} \cdot \Delta x$$

Equation 1-25.

Since all terms of power Δx^2 and higher have been omitted, the error in the approximation is on the order of Δx^2. This situation is shown in Figure 1-16, where the error increases with increasing values of $|\Delta x|$. Note that the approximation would be exact if the original function were linear.

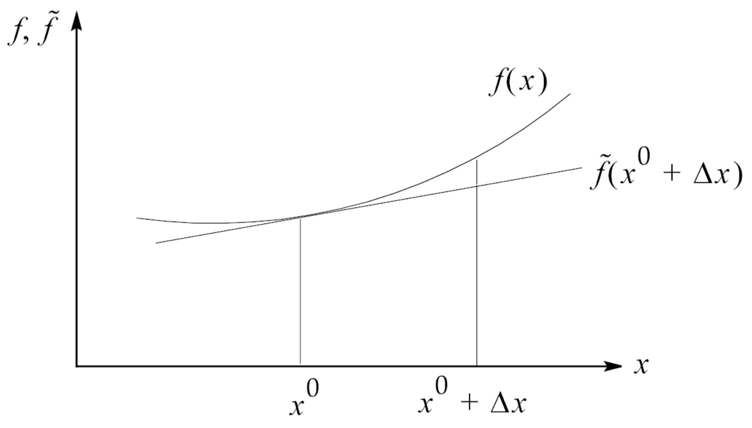


Figure 1-16. Errors in Approximating Functions

In design optimization, we are concerned not just with a single independent variable but rather with a vector of design variables, $\vec{x}$. Under this condition, the approximations for the objective and constraint functions become

$$\tilde{F}\left(\vec{x}^0 + \Delta\vec{z}\right) = F\left(\vec{x}^0\right) + (\nabla F)_{\vec{x}^0} \cdot \Delta\vec{x}$$

$$\tilde{g}_j\left(\vec{x}^0 + \Delta\vec{x}\right) = g_j\left(\vec{x}^0\right) + (\nabla g_j)_{\vec{x}^0} \cdot \Delta\vec{x}$$

Equation 1-26.

where a gradient term replaces the first derivative term of Eq. 1-25.

We have not yet addressed the issue of how to determine the first derivative, or gradient, information. This information comes from the design sensitivity analysis. We introduce design sensitivity for linear static analysis here but reserve the discussion of the other analysis disciplines for Design Modeling for Sensitivity and Optimization.

Many of our constraints, and perhaps the objective function as well, are based on responses that depend on the solution of the static equilibrium equations:

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

Equation 1-27.

For example, we may have an upper limit constraint on stress or

$$g(\vec{x}) = \sigma_j - \sigma_j^{max} \le 0$$

Equation 1-28.

The stress response σ_j is not only a function of the displacement solution {*u*} but also the element geometry and elastic properties, or

$$\{\sigma\} = [D][B]\{u\}$$

Equation 1-29.

where [*D*]*B*] is the stress-displacement transformation matrix. σ_j is a component of the element stress vector {*σ*}.

Partial differentiation of the stress constraint *g* with respect to the i-th design variable and use of the chain rule for differentiation yields

$$\frac{\partial g}{\partial x_i} = \frac{\partial g}{\partial \sigma_j} \cdot \frac{\partial \sigma_j}{\partial x_i}$$

Equation 1-30.

where ($\partial\sigma_j$ / ∂x_i) is referred to as a sensitivity coefficient. In general, a sensitivity coefficient is defined as the partial derivative of a response with respect to a design variable, or

$$\frac{\partial r_j}{\partial x_i}$$

Equation 1-31.

where r_j is a general response quantity. Now,

$$\frac{\partial r_j}{\partial x_i} = \frac{\partial r_j}{\partial \{u\}} \cdot \frac{\partial \{u\}}{\partial x_i}$$

Equation 1-32.

The first term of Eq. 1-32 is easily determined from relations such as Eq. 1-29. The second term can be evaluated if we first differentiate the static equilibrium equation (Eq. 1-27) with respect to a design variable to obtain

$$[K]\frac{\partial u}{\partial x_i} = \frac{\partial P}{\partial x_i} - \frac{\partial [K]}{\partial x_i}\{u\}$$

Equation 1-33.

This relation can be solved for ($\partial\{u\}$ / ∂x_i) to provide the information necessary to construct the first order approximations for the objective and retained constraints from Eq. 1-26. The solution of Eq. 1-33 is relatively inexpensive given that we already have the stiffness matrix [*K*] available in decomposed form as a result of static analysis.

A Simple Linear Design Space

At this point, it is probably worthwhile to take a look at a linearly approximated design space just to gain a qualitative understanding of the nature of the approximation. For this example, we can again refer to the simple cantilever beam of Numerical Optimization Basics, Figure 1-6.

Recall that the design variables for the cantilever beam are the base B and height H of the beam cross section. If we construct linear approximations to the objective and constraint equations according to Eq. 1-26, we have

$$\tilde{V}(B^0 + \Delta B, H^0 + \Delta H, L) = V(B^0, H^0, L) + \left.\frac{\partial V}{\partial B}\right|_{B^0, H^0} \cdot \Delta B + \left.\frac{\partial V}{\partial H}\right|_{B^0, H^0} \cdot \Delta H$$

$$\tilde{\sigma}(B^0 + \Delta B, H^0 + \Delta H, L) = \sigma(B^0, H^0, L) + \left.\frac{\partial \sigma}{\partial B}\right|_{B^0, H^0} \cdot \Delta B + \left.\frac{\partial \sigma}{\partial H}\right|_{B^0, H^0} \cdot \Delta H$$

$$\tilde{\delta}(B^0 + \Delta B, H^0 + \Delta H, L) = \delta(B^0, H^0, L) + \left.\frac{\partial \delta}{\partial B}\right|_{B^0, H^0} \cdot \Delta B + \left.\frac{\partial \delta}{\partial H}\right|_{B^0, H^0} \cdot \Delta H$$

Equation 1-34.

For an initial design at (6,45), the derivatives in the above equation can be evaluated to yield

$$\tilde{V}(B^0 + \Delta B, H^0 + \Delta H, L) = 1.35 \times 10^5 + 2.25 \times 10^4 \Delta B + 3.0 \times 10^3 \Delta H$$
$$\tilde{\sigma}(B^0 + \Delta B, H^0 + \Delta H, L) = 555.56 - 92.593 \Delta B - 24.691 \Delta H$$
$$\tilde{\delta}(B^0 + \Delta B, H^0 + \Delta H, L) = 2.0576 - 0.34294 \Delta B - 0.13717 \Delta H$$

Equation 1-35.

The design space resulting from Eq.1-35 is shown in Figure1-17 superimposed on the true design space. From the graph, note that the optimum resulting from this linear approximation happens to be at a slightly smaller value than the true objective. The linear approximation has allowed the design to become slightly violated when compared to the true design space.

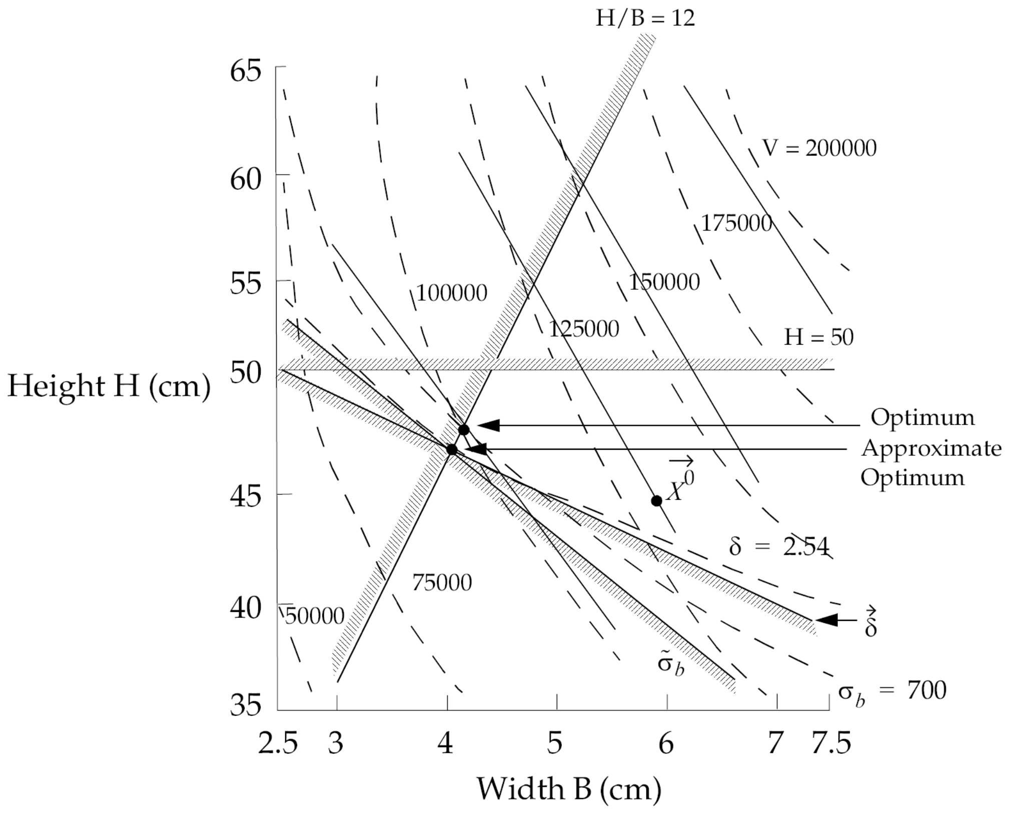


Figure 1-17. Linearly Approximated Cantilever Beam Design Space

Though not yet an optimal design, the approximate optimum is a useful starting point for the next approximate optimization cycle. This next cycle proceeds in exactly the same fashion; an analysis is performed to determine the response values along with an evaluation of the response derivatives. This information is then used to create another approximate subproblem that should, in turn, yield an even better approximation of the true optimum. After only a few more cycles, we may have reached a point sufficiently close to the true optimum that the process converges as measured by the Kuhn-Tucker conditions.

Summary

By now you should have a fairly good idea of some of the various applications of design optimization as well as some of the numerical optimization capabilities available to assist you in your design tasks. We introduced the basic ideas of numerical optimization and some of the methods that are used to couple it with structural analysis codes. Central to this has been the recognition that it is impractical to directly link numerical optimizers and structural analysis codes to create a structural optimizer, and that some sort of approximations are required.

It has also been shown that the set of approximation concepts acts as an interface between the analysis and the optimizer. There is really nothing extraordinary about these approximations since in many ways they are similar to the methods experienced designers already use. Of course, formalizing these methods in structural optimization allows problems of considerably greater size and complexity to be solved than with traditional design methods alone.

Chapter 2: Design Modeling for Sensitivity and Optimization

A design model is required for design sensitivity and optimization in much the same way an analysis model is required for finite element analysis. The design model is simply a formal statement of allowable changes that can be made to a structure during the search for an optimal design. It also places limits on these allowable changes, and defines limits on the structural responses. Constructing the design model requires good engineering judgement if the design results are to be of any practical use.

A design model must:

- Define the design variables that may be modified.
- Describe the relationship between the design variables and the analysis model properties and/or grid locations (for shape optimal design).
- Define the objective function, which provides a scalar measure of design quality.
- Place bounds (constraints) limiting the design responses to an acceptable range.

This chapter describes each of these features in greater detail. In connection with this chapter, you may want to refer to Input Data, as well as Example Problems. These examples should all be available on your system. Running some of these in connection with the following discussions is a good way to begin to discover the many design modeling options available to you in NX Nastran.

2.1 Overview of Design Modeling

Given the wide range of problem types that may be addressed using design sensitivity and optimization, the process of generating a design model is hardly a rote or mechanical operation. However, most of the operations are sufficiently similar that one might generate a flowchart outlining the design modeling process as follows:

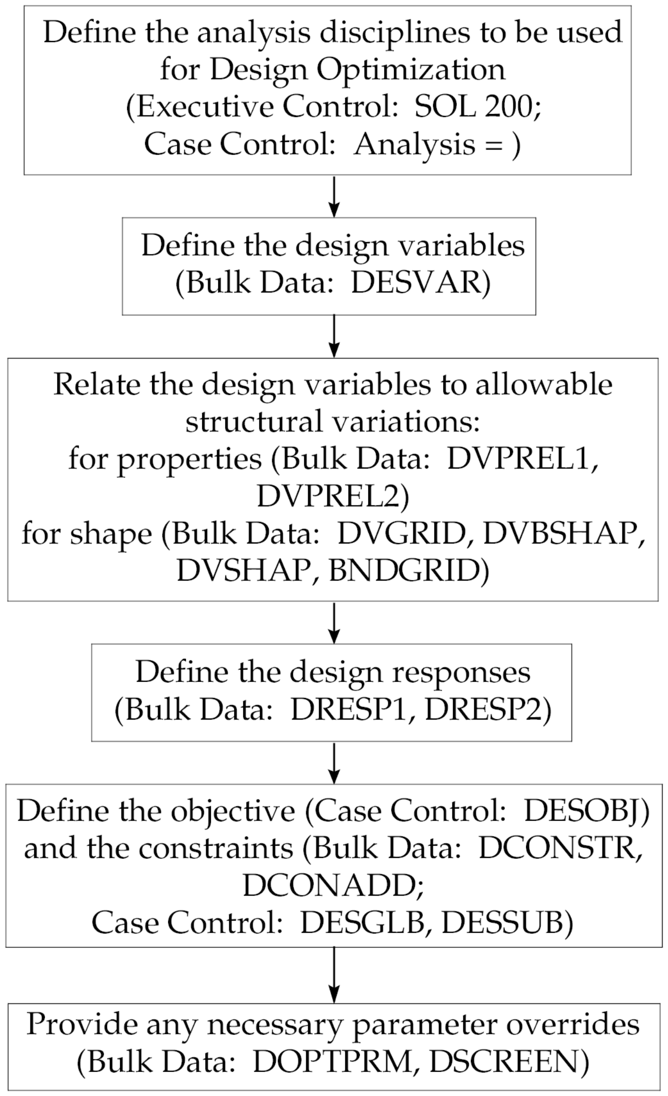

Define the analysis disciplines to be used for Design Optimization (Executive Control: SOL 200; Case Control: Analysis =)
Define the design variables (Bulk Data: DESVAR)
Relate the design variables to allowable structural variations: for properties (Bulk Data: DVPREL1, DVPREL2) for shape (Bulk Data: DVGRID, DVBSHAP, DVSHAP, BNDGRID)
Define the design responses (Bulk Data: DRESP1, DRESP2)
Define the objective (Case Control: DESOBJ) and the constraints (Bulk Data: DCONSTR, DCONADD; Case Control: DESGLB, DESSUB)
Provide any necessary parameter overrides (Bulk Data: DOPTPRM, DSCREEN)

Figure 2-1. Design Modeling Process

The following sections discuss each of these items in greater detail. (Overriding optimization control parameters with the DOPTPRM and DSCREEN entries is reserved until after approximation concepts have been presented in Example Problems.)

2.2 Defining the Analysis Disciplines

A powerful feature of design sensitivity and optimization in NX Nastran is that you can specify that a structure be subjected to a number of different analysis types for a number of various load conditions. The optimizer will consider the results of all of these analyses simultaneously when proposing an improved design. This approach is often described as multidisciplinary design optimization, and is the only rational way of proposing a useful optimal design.

For example, you may have a component that is subjected to two static loading conditions. The component must also satisfy requirements on natural frequency bounds as determined from a modal analysis. Further, the structure might also be subjected to transient loading conditions for which peak displacement responses might be of concern. We could quite easily introduce these different analysis types in Solution 200 (the solution sequence for design optimization) with the following input:

```
SOL 200
cend
spc = 100
DESOBJ(MIN) = 15
ANALYSIS = STATICS
subcase 1

DESSUB = 10
 displacement = all
 stress = all
 load = 1
subcase 2

DESSUB = 20
 displacement = all
 strain(fiber) = all
 load = 2
subcase 3

ANALYSIS = MODES
 DESSUB = 30
 method = 3
subcase 4

ANALYSIS = MTRAN
 DESSUB = 40
 method = 4
 dload = 4
begin bulk
```

Note some of the features in this example:

SOL 200	Executive command indicating Solution 200, the solution sequence for design sensitivity and optimization, is to be invoked.
ANALYSIS	Case Control command indicating the analysis discipline to be used for a particular subcase (if appearing above a subcase level, all subsequent subcases assume the same ANALYSIS command until changed). ANALYSIS may assume any of the following values:

STATICS	Statics
MODES	Normal Modes
BUCK	Buckling
DFREQ	Direct Frequency Response
MFREQ	Modal Frequency Response
MTRAN	Modal Transient
SAERO	Steady Aeroelastic
FLUTTER	Flutter

DESOBJ	Selects the design objective from a response defined in Bulk Data with DRESP1 or DRESP2.

DESSUB	Selects constraints to be applied at a particular subcase level from a DCONSTR set defined in Bulk Data.

DESOBJ and DESSUB are discussed in greater detail in Glossary of Terms.

Case Control Output Requests in Design Optimization

In addition to the analysis-related Case Control commands, output requests might also appear, such as:

```
displacement= all
strain(fiber)= all
```

Note

The frequency of data recovery output (e.g. first design cycle, last, every n-th, etc.) is governed by the Bulk Data parameter NASPRT. See Parameters for Design Sensitivity and Optimization for details.

These are of interest to design optimization in a couple of ways:

- In general, case control requests are for data recovery only; strictly speaking, they are unnecessary for design optimization. All necessary data recovery is automatically performed based on the design responses identified in the Bulk Data (identifying the responses is discussed in Identifying the Design Responses. You do not need to explicitly request this data recovery unless you want to view the results.

- If various output forms are possible (e.g., strain(strcur) or strain(fiber)), design optimization will use the defaults unless a Case Control request to the contrary is provided. To use the preceding example, STRAIN(STRCUR) representation is the default. If STRAIN(FIBER) is provided, data recovery and design optimization responses are both computed using this form. Defaults are listed in the *NX Nastran Quick Reference Guide*.

Shape optimization and superelement optimization each have some additional Case Control considerations beyond those related to data recovery. Superelement Design Modeling discusses the Case Control structure for superelement design models. In addition, Example Problems includes examples of both shape and superelement optimization.

2.3 Defining the Design Variables

In design optimization, design variables are the quantities that are modified by the optimizer during the search for an improved design. In design sensitivity analysis, rates of change of responses are computed with respect to the design variables.

Defining the design variables is part of the design modeling task of the engineer. Once defined, the design variables may be used to

- Describe analysis model property variations.

- Define shape variations.

Design variables are often thought of as plate thicknesses, radii of cutouts, etc. Of course, design variables may be used to describe these quantities, but the idea is actually much more general and powerful.

Design Variable Generality

As the optimizer changes the design variables, it does not "know" it is changing a structural model – it is only performing a mathematical search. It is the design engineer who has defined exactly how the structure will change. For example, a single design variable might have been used to describe a number of bar cross-sectional areas, several plate element thicknesses as well as a fillet radius. As the optimizer changes this single variable, the entire structure changes! Expressing allowable structural variations with an economy of design variables is part of the challenge of design modeling.

DESVAR Bulk Data Entry

A Design Variable is defined with a DESVAR Bulk Data entry. From the Bulk Data listings in Input Data, notice that the DESVAR entry provides an ID, an initial value for the variable, and bounds (or side constraints). These bounds limit the region of search with respect to each variable as

$$x_i^L \le x_i \le x_i^U$$

Equation 2-1.

The optimizer will never propose a design that violates these bounds. (See also the "Selective Modification of Design Variable Bounds" in the Relating Design Variables to Properties section.)

The following DESVAR entries define a pair of variables x_{10} and x_{11} with initial values of 0.05 and 0.03, respectively.

```
$DESVAR, ID,      LABEL,  XINIT,  XLB,     XUB,     DELXV
$
DESVAR, 10,      AREA1,  0.05,   0.01,   0.1
DESVAR, 11,      THICK1, 0.03,   0.01,   0.08
$
```

From the label fields, we can assume these design variables will be used to define an area and a thickness, although the DESVAR entry does not actually provide these relations (this is up to the DVPREL1 and DVPREL2 entries). Bounds have also been defined as

$$0.01 \le x_{10} \le 0.1$$
$$0.01 \le x_{11} \le 0.08$$

Equation 2-2.

The DELXV field, which has been left blank here, limits the amount the design variable can change during an optimization cycle; by default, this amount is 100%.

Design Variables and the Dimension of the Design Space

The entire set of DESVAR entries can be thought of as defining a vector of design variables or

$$\vec{x} = [x_1, x_2, \ldots, x_n]^T$$

Equation 2-3.

Note

To date, optimization problems that involve over several thousand design variables have been efficiently solved using NX Nastran.

The quantity *n* is often referred to as the dimension of the design space. The greater the dimension the more complicated the optimization task. The optimizer's task is to find the best configuration for these variables.

Design Variables and the Basic Optimization Problem Statement

Recall that the problem we are trying to solve is defined by the basic optimization problem statement, Eq. 2-1 through Eq. 2-5. Note that the design objective and constraint functions are expressed as functions of design variables. How is this the case in structural optimization?

Assume that we are interested in minimizing the total structural mass for a particular mechanical component. This will be our objective function. We know that the mass is a function of the properties used to describe the component as well as its shape. This can be expressed as

$$M = M(\vec{p}, \vec{G})$$

Equation 2-4.

where the vector $\vec{p}$ is the collection of properties that describe the model and the vector $\vec{G}$ is the collection of grid point coordinates.

In the next sections, we will show how the properties and shape of the structure can be expressed as functions of design variables or

$$\begin{aligned} \vec{p} &= \vec{p}(\vec{x}) \\ \vec{G} &= \vec{G}(\vec{x}) \end{aligned}$$

Equation 2-5.

By direct substitution of Eq. 2-5 into Eq. 2-4,

$$M = M(\vec{p}(\vec{x}), \vec{G}(\vec{x}))$$

Equation 2-6.

or simply,

$$M = M(\vec{x})$$

Equation 2-7.

To summarize, as the optimizer changes the design variables, the analysis model properties and shape will also change as defined by the design model. The modified properties and shape result in changes to the computed responses that we have used to define the objective and constraint functions. Based on the modified objective and constraints, the optimizer can measure the amount of design improvement.

2.4 Relating Design Variables to Properties

For the analysis model to vary as the design variables are changed, its properties and/or its shape must be expressed in terms of the design variables. You need to provide these relations as part of the design model specification. This section discusses analysis model property changes. Relating Design Variables to Shape Changes discusses shape variations.

Definition of Analysis Model Properties

Analysis model properties are quantities that appear on bulk data property entries. Plate thicknesses, area moments of inertia, elastic spring stiffnesses, elastic modulii, coefficients of thermal expansion, grid point locations, and so on, are all examples of analysis model properties. Analysis model properties can be written as functions of design variables. Similarly, connectivity properties and material properties can be written as functions of design variables. For additional information on properties that can be written as a function of design variables, see the DVCREL1, DVCREL2, DVGEOM, DVGRID, DVMREL1, DVMREL2, DVPREL1, and DVPREL2 bulk data entries in the *NX Nastran Quick Reference Guide*.

Design Variable-to-Property Relation Types

There are two ways to relate design variables to properties.

- Type-1 relations define a linear relationship between design variables and properties.
- Type-2 relations use the equation input capability of NX Nastran to define a functional relationship between design variables and properties.

Type-1 Design Variable-to-Property Relations

Type-1 design variable-to-property relations define a linear relationship between the design variables and properties. DVCREL1, DVMREL1, and DVPREL1 bulk entries are used to define Type-1 design variable-to-property relations.

- DVCREL1 (Design-Variable-Connectivity RELation, type-1) is used to relate design variables to connectivity properties.
- DVMREL1 (Design-Variable-Material RELation, type-1) is used to relate design variables to material properties.
- DVPREL1 (Design-Variable-Property RELation, type-1) is used to relate design variables to analysis model properties.

Type-1 (linear) design variable-to-property relations are of the form

$$p_j = C_o + \sum_i C_i x_i$$

Equation 2-8.

where p_j is the j-th property, expressed as a linear combination of the design variables.

Type-2 Design Variable-to-Property Relations

Type-2 design variable-to-property relations define a functional relationship between design variables and properties. The functional relationships are defined on DEQATN (Design EQuATioN) bulk entries. DVCREL2, DVMREL2, and DVPREL2 bulk entries that reference DEQATN bulk entries are used to define a Type-2 design variable-to-property relations.

- DVCREL2 (Design-Variable-Connectivity RELation, type-2) is used to relate design variables to connectivity properties.
- DVMREL2 (Design-Variable-Material RELation, type-2) is used to relate design variables to material properties.
- DVPREL2 (Design-Variable-Property RELation, type-2) is used to relate design variables to analysis model properties.

Type-2 design variable-to-property relations are of the form

$$p_j = f(\{x\}, \{c\})$$

Equation 2-9.

where p_j is the j-th property, expressed as a function of a collection of design variables and constants.

The same design variables may appear in both Type-1 and Type-2 relations simultaneously. This is depicted in Figure 2-2 where the design variable *x* appears in both DVPREL1 and DVPREL2 bulk entries.

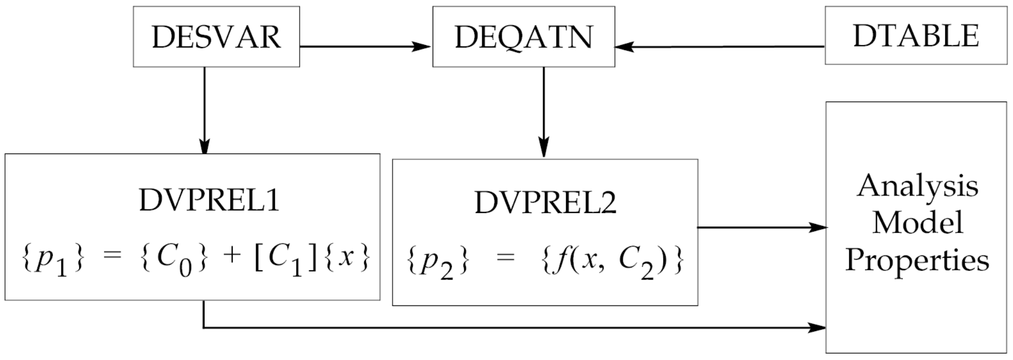

DESVAR
DEQATN
DTABLE
DVPREL1
$\{p_1\} = \{C_0\} + [C_1]\{x\}$
DVPREL2
$\{p_2\} = \{f(x, C_2)\}$
Analysis Model Properties

Figure 2-2. Design Variable-to-Property Relations

The DVPREL1 bulk entry defines linear property variations and requires only design variable input from DESVAR bulk entries. All necessary constants are supplied directly on the DVPREL1 bulk entry. Thus, Figure 2-2 shows the DVPREL1 block with a single input and a single output.

The DVPREL2 bulk entry relies on an equation to describe the (generally) nonlinear property variations. The input to this equation can include both design variable input from DESVAR bulk entries and table constant input from DTABLE bulk entries. Thus, Figure 2-2, shows the DEQATN block with two possible inputs and a single output, and the DVPREL2 block with a single input and a single output.

Selective Modification of Design Variable Bounds

Many practical problems have properties that are linearly related to a single design variable. An example is where the thickness of each ply of a composite shell element is proportional to the thickness of all other plies of the same element. For such cases, if both the properties and the design variable are bounded, NX Nastran will automatically remove bounds on the properties by transferring more restrictive bounds to the related design variable.

For example, consider a single property, *p*, and a design variable, *x*, that are linearly related as follows:

$$p = 3.0 + 2.0x$$

Equation 2-10.

Solving for *x* yields:

$$x = (p - 3.0)/2.0$$

Equation 2-11.

Suppose the bounds for *p* and *x* are specified to be:

$$10.0 \le p \le 30.0$$

Equation 2-12.

and

$$2.0 \le x \le 9.0$$

Equation 2-13.

Applying the property bounds to the above equation for *x* gives:

$$3.5 \le x \le 13.5$$

Equation 2-14.

The new lower bound of 3.5 is more restrictive than the specified lower bound of 2.0. Thus, the bounds for *x* are modified such that:

$$3.5 \le x \le 9.0$$

Equation 2-15.

and the bounds on the property *p* are removed.

If an even more restrictive value for the lower bound of the same *x* is found from another property, then the bound is further modified.

You can optionally disable selective modification of design variable bounds with system cell 539.

When design variable bounds are modified as above, the current default for property move limit of DELP = 0.2 may be too conservative. A minimum value of DELP = 0.5 is recommended. Using DELP = 0.5 may improve solution efficiency in other cases as well. The DELP design optimization parameter is specified on the DOPTPRM bulk entry.

Reference by Property Entry

In design variable-to-property relations, analysis model properties are referenced by property entry, rather than on an element-by-element basis. This relation is shown schematically in Figure 2-3. The property IDs PID1 through PIDm are referenced on the DVPREL1 and DVPREL2 entries. Since each of these property groups may contain a large number of elements (EID1 through EIDn), you can control many elements by using only a small set of DVPREL-type entries. For example, to modify plate thicknesses, all that is required is to reference a PSHELL ID on a DVPREL bulk entry. All elements in the property group will then change accordingly.

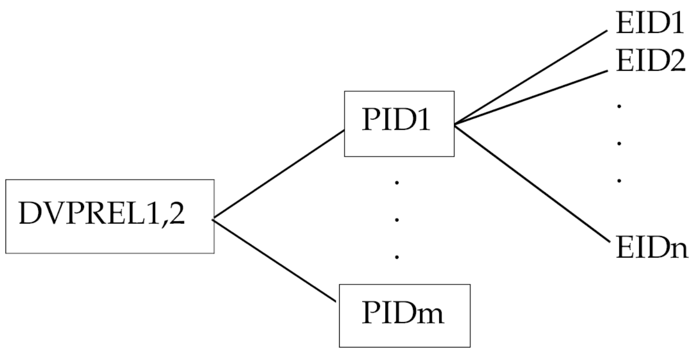


Figure 2-3. Design Model Reference by Property Entry

Identification of Property Entry Items

You must identify a selected property entry item either by its field position on the property Bulk Data entry or by its word position in the element property table (EPT) that is generated by NX Nastran. In general, it is far easier to refer to the field position on the Bulk Data entry. In some instances, (BEAM elements in particular) reference to the element property table must be made instead. None of this is really as difficult as it sounds and is explained in the section on special modeling considerations at the end of this section.

To illustrate both type-1 and type-2 design variable-to-property relations, consider the stiffened panel section of Figure 2-4. The example is test problem library example number D200X7, presented in Stiffened Plate. Three design variables are each related to the plate thickness, web thickness, and cross-sectional area of the web cap, respectively. By grouping all of the elements in the base plate into one property group, their thickness can all be controlled by a single design variable. This same grouping is also used for the elements of the web as well as for the cap section elements.

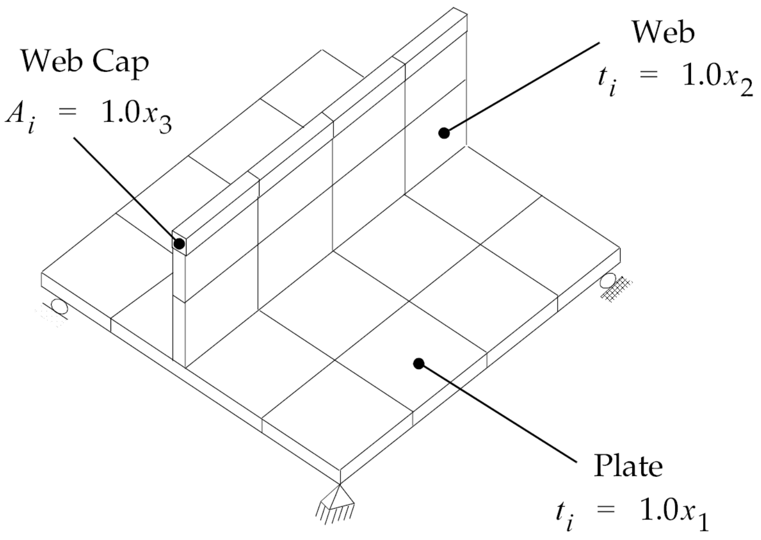


Figure 2-4. Stiffened Panel Test Problem

The base plate thickness is linearly related to a design variable according to

$$t_{100} = 1.0x_1$$

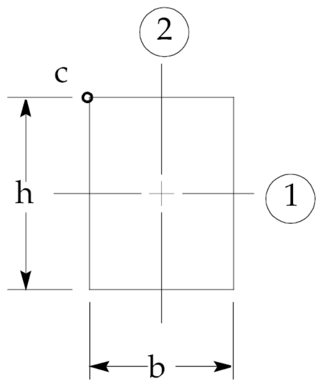
Equation 2-16.

This linear relation can be defined with a DVPREL1 Bulk Data entry as follows:

```
PSHELL, 1,      1,      0.15,   1
$
$...Define the design variables:
$
$DESVAR,ID,     LABEL,  XINIT,  XLB,    XUB,    DELXV
DESVAR, 1,      T-PLATE,0.15,   0.001,  10.0
$
$...Relate the design variables to analysis model properties
$   (linear relations, so use DVPREL1)
$
$DVPREL1,ID,    TYPE,   PID,    FID,    PMIN,   PMAX,   C0,     ,       +
$+,     DVID1,  COEF1,  DVID2,  COEF2,  ...
DVPREL1,1,      PSHELL, 1,      4,      0.01,   ,       ,       ,       +DP1
+DP1,   1,      1.0
```

The PSHELL 1 entry defines the analysis model property group (to which a number of elements belong). The DESVAR 1 entry defines design variable 1 (T-PLATE) with an initial value of 0.15 and lower and upper bounds of .001 and 10.0, respectively. Design variable 1 is, in turn, related to PSHELL property group 1 thickness via DVPREL1 number 1. The 4 in DVPREL1 number 1, field 5, states that it is field 4 on the property entry (the thickness) that is to be varied. The continuation line on the DVPREL1 entry gives the functional relationship defined by Eq. 2-16. Field 6 on the DVPREL1 entry defines the minimum allowable value of the property; here, a minimum thickness of 0.01. This value enforces a lower bound, even though greater than the design variable's lower bound. See Stiffened Plate for the remainder of the design model.

Frequently, you may want to specify the bar cross-sectional dimensions in the design model instead and use this data to compute the resultant cross-sectional properties, such as areas and moments of inertia. Since these properties are often nonlinear functions of the design variables, this approach requires the use of type-2 design variable to property relations. Probably the easiest way to introduce type-2 relations is with a simple example:



$$b = 0.3$$
$$h = 0.4$$
$$I_1 = \frac{bh^3}{12}$$
$$I_2 = \frac{hb^3}{12}$$

Figure 2-5. Cap Section Detail

Assume that the cap section of the stiffened plate of Figure 2-4 is a rectangular section bar as shown in Figure 2-5. Suppose the bar is characterized by the cross-sectional dimensions *b* and *h* and is oriented with respect to principal planes 1 and 2 as shown. A stress recovery location, such as point *c,* might be defined as well. The Bulk Data describing the analysis and design model relations could be written as follows:

```
$...ANALYSIS MODEL PROPERTIES:
PBAR,   120,    220,    0.12,    1.6E-3,    9.0E-4,       ,       ,       ,       +
+,      -.15,   .2
$
$...BAR CROSS-SECTIONAL DIMENSIONS:
$ESVAR, ID,     LABEL,  XINIT,  XLB,    XUB,    DELXV
DESVAR, 10,     B,      .3,     .1,     1.0
DESVAR, 11,     H,      .4,     .1,     1.0
$
$...BAR PROPERTY RELATIONS, A, I1, I2:
$DVPREL2,ID,    TYPE,   PID,    FID,    PMIN,   PMAX,   EQID,   ,       +
$+,      DESVAR,DVID1,  DVID2,  ...,    ,       ,       ,       ,       +
$+,      DTABLE,CID1,   CID2,   ...
DVPREL2,250,    PBAR,   120,    4,      ,       ,       ,501,    ,       +
+,      DESVAR, 10,     11
DVPREL2,251,    PBAR,   120,    5,      1.0E-5  ,       ,502,    ,       +
+,      DESVAR, 10,     11
DVPREL2,252,    PBAR,   120,    6,      1.0E-5  ,       ,503,    ,       +
+,      DESVAR, 10,     11
$
$...EQUATIONS:
DEQATN  501     AREA(B,H) = B*H
DEQATN  502       I1(B,H) = B*H**3/12.
DEQATN  503       I2(B,H) = H*B**3/12.
$
$...STRESS RECOVERY POINT LOCATIONS:
$DVPREL1,ID,    TYPE,   PID,    FID,    PMIN,   PMAX,   C0,     ,       +
$+,     DVID1,  COEF1,  DVID2,  COEF2,  ...
DVPREL1,260,    PBAR,   120,    12,     -.5     ,       ,       ,       +
+,      10,     -.5
DVPREL1,261,    PBAR,   120,    13,     .05,    ,       ,       ,       +
+,      11,     .5
```

PBAR 120 defines the bar properties corresponding to the initial cross-sectional dimensions. Note that the initial values of the design variables listed on the DESVAR entries agree with the initial properties. If any discrepancies exist between the initial analysis model properties (given on the property entries) and those calculated based on the initial values of the design variables, the properties based on the design model have precedence. The design model data is then used to override the analysis model properties. In this example, the design and analysis model agree, and no overrides take place.

Since the bar area and moments of inertia are nonlinear in the design variables, they must be defined using equations. These relations are defined on the DEQATN entries 501, 502, and 503. The input arguments are assigned via the DVPREL2 entries 250, 251, and 252. The DESVAR identifier on the first continuation of each of these entries indicates design variables 10 and 11 are to be used as input (*b* and *h*) for each of the referenced equations.

Note

Override of analysis model data by design model data provides a convenient method of restarting in design optimization. See the restart example in Restarts in Design Optimization.

Equation Arguments

The input arguments defined on the DEQATN entries are formal arguments, which are defined at run time by the actual arguments provided on the DVPREL2 entries. The rules for defining these equations follow directly from FORTRAN syntax requirements. However, all arguments are assumed to be real numbers. FORTRAN intrinsic functions may also be used. (Some of these functions, such as ABS and MAX, are not recommended since they may yield functions having discontinuous first derivatives. The sensitivity analysis as well as the optimization results may be incorrect if any of these discontinuities are encountered.)

Note

The available intrinsic functions and their uses are listed with the DEQATN bulk data entry in the *NX Nastran Quick Reference Guide*.

Minimum Allowable Property Values

Field 6 on the DVPREL2 entries defines PMIN, which is the minimum allowable value for the particular property during design optimization. Generally, its purpose is to prevent properties from taking on values near zero. The default is 0.001, which is acceptable for many applications. Here, I_2 is already less than this default, and I_1 may take on values less than the default as well (based on the minimum B and H values). Therefore, we override the default in both cases and specify a PMIN of 1.0E-5 on DVPREL2 251 and 252, based on engineering judgment.

A similar situation is seen in the DVPREL1 relations that in this case, relate the location of the stress recovery point *c* to the values of the design variables. The *y*-coordinate location is a negative quantity, but the default PMIN for stress recovery locations is -1.0E+35 (see the DVPREL1 and DVPREL2 Bulk Data entry descriptions). Just to be explicit, a negative PMIN has been provided on DVPREL1 260. DVPREL1 260 and 261 respectively, define the following linear relations:

$$\begin{aligned} C_y &= -b/2 \\ C_z &= h/2 \end{aligned}$$

Equation 2-17.

Stress Recovery Locations: Words of Warning

Note the error that may have resulted if the location of the stress recovery point had not been allowed to vary. Since the stress computations would not have included the direct effects of changes in the stress recovery point, the stresses seen by the optimizer would not be those that the engineer had intended. This would certainly lead to incorrect results. Of course, other stress recovery points would probably also need to be defined to ensure recovery of the maximum stresses.

Special Design Modeling Considerations - BEAM Elements

In design modeling, the BEAM element differs from other element types in that its properties must be referenced according to their word positions in the element property tables. The element property tables are internally generated in NX Nastran, and used to form the element-level structural matrices.

Note

By convention, field position identifiers are denoted as positive quantities on DVPRELi entries, EPT word positions are denoted as negative quantities.

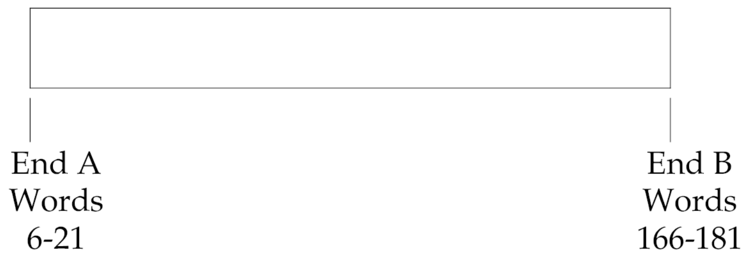
The source of this difference is due to the generality of the PBEAM Bulk Data entry. Since a single entry can used to specify the BEAM's cross-sectional properties for both ends and up to nine intermediate stations, a unique correspondence between a BEAM property and its field position on the PBEAM entry is not assured. The only relation that is assured is the one between a property item and its word position in the element property table. This correspondence requires that the engineer have some understanding of the structure of the EPT for the BEAM element and its relation to the data supplied on the PBEAM entry.

Note

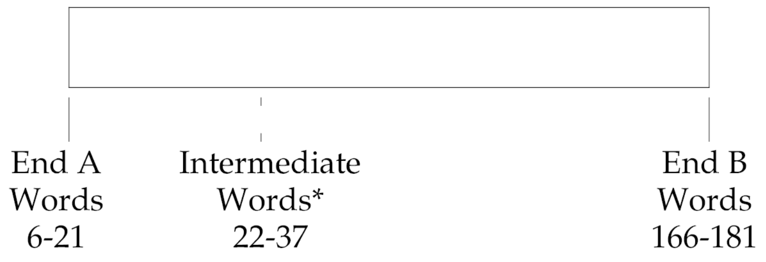
Note that the numbering in Figure 2-6 applies to the "new" design sensitivity and optimization (Solution 200).

In general, the cross section of a BEAM element may be either constant, linear, or variable. These three possibilities are shown in Figure 2-6, which is a schematic of the EPT configuration for each of these situations.

1. Constant section beam (only end A specified): Words 6-21 and 166-181 must be accounted for on DVPRELi entries.

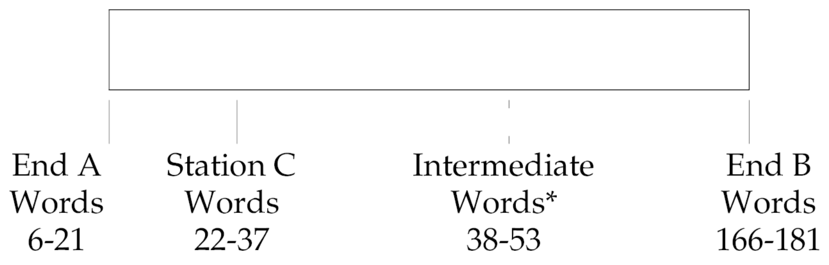


2. Constant or variable section beam (end A and end B specified): Words 6-21, 22-37, and 166-181 must be accounted for on DVPRELi entries.



*This station is internally generated but must be accounted for on DVPRELi entries. Words 22-37 are identical to Words 166-181 (End B).

3. Constant or variable section beam (end A, intermediate station C and end B specified). (This case is typical of any number of intermediate stations up to and including 8.) Words 6-21, 22-37, 38-53, and 166-181 must be accounted for on DVPRELi entries. Thus, every intermediate station plus one internally generated intermediate station must be accounted for on the set of DVPRELi entries.



*This station is internally generated but must be accounted for on DVPRELi entries. Words 38-53 are identical to Words 166-181 (End B).

Figure 2-6. Element Property Table for BEAM Element

Item 1 in Figure 2-6 is a diagram of the EPT for the BEAM element when the PBEAM entry furnishes data at end A only. End B data is built internally as a copy of end A properties. Since this structure is used by subsequent modules to generate the element mass and stiffness matrices, the design model must also define equal variations for both sets of data. Thus, for design sensitivity and optimization, you must prescribe EPT word positions 166-181 (end B) in addition to words 6-21 (end A). Otherwise, the ensuing incompatibility between end A and end B data will invalidate the subsequent analysis results.

Item 2 in Figure 2-6 shows the structure of the EPT for a tapered beam. The data for ends A and B is determined based on the supplied PBEAM input. Additionally, the first intermediate station (Words 22-37) is internally generated and contains a copy of end B data. Your design model must then reference words 6-21 for end A, words 166-181 for end B, and the first intermediate station.

The most general case is shown in item 3 where, in addition to ends A and B, from one to nine intermediate stations are defined. As in item 2, the first available intermediate station always contains a copy of end B data. This must always be accounted for in the design model definition. (Note that if nine intermediate stations are defined, there is no need to supply an additional copy of end B data since the next station is end B itself.)

To illustrate, consider the uniformly tapered beam element in Figure 2-7. The cross-sectional dimensions for a rectangular element can be described in terms of design variables b_1, h_1, b_2, and h_2 as shown in the figure. An input data file using this element in a single-element design sensitivity test is given in Listing 2-1. This example corresponds to a test of Item 2 in Figure 2-6.

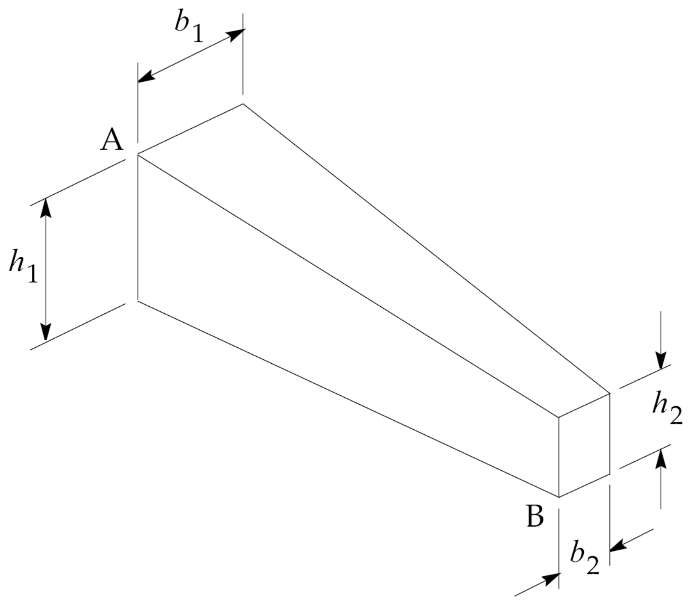


Figure 2-7. Tapered Rectangular Beam Element

The initial design is given by

$$\{x\} = \begin{Bmatrix} b_1 \\ h_1 \\ b_2 \\ h_2 \end{Bmatrix} = \begin{Bmatrix} 1.0 \\ 2.0 \\ 0.5 \\ 1.0 \end{Bmatrix}$$

Equation 2-18.

This design is input to the DEQATN entries to define the corresponding cross-sectional properties for the element.

In addition to the properties at ends A and B, the design model must also include a definition for the first intermediate station. The properties at end A are expressed as functions of Design Variables 1 and 2 via DVPREL2s 1-5, end B properties are expressed in terms of Design Variables 3 and 4 via DVPREL2s 6-10, and the first intermediate station by DVPREL2s 11-15.

In order to make identification of the EPT word positions easier, the PBEAM entry has been set up to define cross-sectional properties that are slightly different from the properties computed using the initial design variable values. The design model overrides the analysis model properties and the differences reported in a differences comparison table. Since the word positions in the EPT are also output, these results can be used to check the validity of the formulation.

Note

The design model should include stress recovery point locations. Serious design errors may result if you omit these from your design model formulation.

The difference comparison table associated with the design model override is shown in Figure 2-8. From the corresponding field IDs (FID) it can be seen that not only is the data for ends A and B in the expected word positions, but the intermediate station is also present as expected. The analysis and design property columns indicate that it is indeed just a copy of the data for end B. Furthermore, the design model formulation appears to be correct judging from the number, location, and values of the design model override data.

```
                ----- COMPARISON BETWEEN INPUT PROPERTY VALUES FROM ANALYSIS AND DESIGN MODELS -----

     --------------------------------------------------------------------------------------------------
      PROPERTY    PROPERTY      FIELD      ANALYSIS         DESIGN         LOWER          UPPER      DIFFERENCE
        TYPE         ID           ID          VALUE          VALUE          BOUND          BOUND          FLAG
     --------------------------------------------------------------------------------------------------
      PBEAM          100        -177    -5.100000E-01  -5.000000E-01  -1.000000E+35   1.000000E+20   WARNING
      PBEAM          100        -175     5.100000E-01   5.000000E-01  -1.000000E+35   1.000000E+20   WARNING
      PBEAM          100        -170     4.200000E-02   4.166667E-02   1.000000E-03   1.000000E+20   WARNING
      PBEAM          100        -169     1.040000E-02   1.041667E-02   1.000000E-03   1.000000E+20   WARNING
      PBEAM          100        -168     5.100000E-01   5.000000E-01   1.000000E-03   1.000000E+20   WARNING
      PBEAM          100         -33    -5.100000E-01  -5.000000E-01  -1.000000E+35   1.000000E+20   WARNING
      PBEAM          100         -31     5.100000E-01   5.000000E-01  -1.000000E+35   1.000000E+20   WARNING
      PBEAM          100         -26     4.200000E-02   4.166667E-02   1.000000E-03   1.000000E+20   WARNING
      PBEAM          100         -25     1.040000E-02   1.041667E-02   1.000000E-03   1.000000E+20   WARNING
      PBEAM          100         -24     5.100000E-01   5.000000E-01   1.000000E-03   1.000000E+20   WARNING
      PBEAM          100         -17    -1.010000E+00  -1.000000E+00  -1.000000E+35   1.000000E+20   WARNING
      PBEAM          100         -15     1.010000E+00   1.000000E+00  -1.000000E+35   1.000000E+20   WARNING
      PBEAM          100         -10     6.670000E-01   6.666667E-01   1.000000E-03   1.000000E+20   WARNING
      PBEAM          100          -9     1.670000E-01   1.666667E-01   1.000000E-03   1.000000E+20   WARNING
      PBEAM          100          -8     2.010000E+00   2.000000E+00   1.000000E-03   1.000000E+20   WARNING
 1. IF FIELD ID IS LESS THAN ZERO, IT IDENTIFIES THE WORD POSITION OF AN ENTRY IN EPT.
 2. IF FIELD ID IS GREATER THAN ZERO, IT IDENTIFIES THE FIELD POSITION ON A PROPERTY BULK DATA ENTRY.
 3. THE DIFFERENCE FLAG IS USED TO CHARACTERIZE DIFFERENCES BETWEEN ANALYSIS AND DESIGN MODEL PROPERTIES:
 IF THE FLAG IS NONE, THEN THERE IS NO SIGNIFICANT DIFFERENCE BETWEEN THE TWO VALUES.
 IF THE FLAG IS WARNING, THEN THE USER IS ADVISED THAT DIFFERENCES EXIST.
 IF THE FLAG IS FATAL, THEN THE DIFFERENCES ARE GREATER THAN  1.00000E+35 AND THE RUN WILL BE TERMINATED.
```

Figure 2-8. Message Indicating an Analysis Model Override by the Design Model

```
$>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
$ BEAM MODELING TEST, VERSION 2
$
$    TAPERED BEAM WITH RECTANGULAR SECTION, DESIGN SENSITIVITY ANALYSIS. SINCE
$    BOTH ENDS A AND B ARE SPECIFIED ON THE PBEAM ENTRY, DESIGN CHANGES MUST
$    BE SPECIFIED NOT ONLY FOR THE ENDS, BUT ALSO FOR THE FIRST INTERMEDIATE
$    STATION, WHICH CONTAINS A COPY OF END B DATA.
$
$
$                      PLANE 2
$                         |
$                     ----*----
$                     |       |
$                  H  |       | --- PLANE 1
$                     |       |
$                     ----*----
$                         B            * =  STRESS RECOVERY LOCATIONS
$
$
$>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>>
$
TIME 10
SOL 200
CEND
$
```

```
$ CC FOR ANALYSIS:
DISP   = ALL
STRESS = ALL
SPC    = 100
LOAD   = 300
$ CC FOR OPTIMIZATION:
ANALYSIS    = STATICS
DESOBJ(MIN) = 8      $ OBJECTIVE FUNCTION DEFINITION
DESSUB      = 10     $ CONSTRAINT SET SELECTION
$
BEGIN BULK
$PARAM, OPTIM, NO
PARAM, OPTEXIT, 4
$------------------------------------------------------------------------------
$ ANALYSIS MODEL
$------------------------------------------------------------------------------
MAT1    110     10.0E6          0.33    0.1                             +M1
+M1     50000.  50000.  29000.
GRDSET                                                  4
GRID    1               0.0     0.0     0.0
GRID    2               20.     0.0     0.0
CBEAM   1       100     1       2       0.0     1.0     0.0
$
$ PBEAM ENTRY INPUT WITH SLIGHT 'ERROR' IN TERMS.  THIS HELPS VALIDATE THE DE-
$ SIGN MODEL BECAUSE USER WARNING MESSAGE WILL BE ISSUED, CONFIRMING OVERRIDE.
$
PBEAM   100     110     2.01    .167    .667                            +P11
+P11    0.0     1.01    0.0     -1.01                                   +P12
+P12    YES     1.0     0.51    .0104   .042                            +P13
+P13    0.0     0.51    0.0     -0.51
$
SPC1    100     123456  1
FORCE   300     2               20000.0 0.0     0.0     -1.0
$------------------------------------------------------------------------------
$ DESIGN MODEL
$------------------------------------------------------------------------------
$
$...END A DATA: (A,I1,I2,C2,D2)
$
$DESVAR,ID,     LABEL,  XINIT,  XLB,    XUB,    DELXV
DESVAR, 1,      B1,     1.0,    0.1,    10.0
DESVAR, 2,      H1,     2.0,    0.2,    20.0

$
$DVPREL2,ID,    TYPE,   PID,    FID,    PMIN,   PMAX,   EQID,   ,       +
$+,     DESVAR, DVID1,  DVID2,  ...,    ,       ,       ,       ,       +
$+,     DTABLE, CID1,   CID2,   ...
DVPREL2,1,      PBEAM,  100,    -8,     ,       ,       101,    ,       +
+,      DESVAR, 1,      2
DVPREL2,2,      PBEAM,  100,    -9,     ,       ,       102,    ,       +
+,      DESVAR, 1,      2
DVPREL2,3,      PBEAM,  100,    -10,    ,       ,       103,    ,       +
+,      DESVAR, 1,      2
DVPREL2,4,      PBEAM,  100,    -15,    ,       ,       104,    ,       +
+,      DESVAR, 2
DVPREL2,5,      PBEAM,  100,    -17,    ,       ,       105,    ,       +
+,      DESVAR, 2
$
$...END B DATA: (A,I1,I2,C2,D2)
$
$DESVAR,ID,     LABEL,  XINIT,  XLB,    XUB,    DELXV
DESVAR, 3,      B2,     0.5,    0.05,   10.0
DESVAR, 4,      H2,     1.0,    0.1,    20.0
$
$DVPREL2,ID,    TYPE,   PID,    FID,    PMIN,   PMAX,   EQID,   ,       +
$+,     DESVAR, DVID1,  DVID2,  ...,    ,       ,       ,       ,       +
$+,     DTABLE, CID1,   CID2,   ...
```

```
DVPREL2,6,      PBEAM,  100,    -168,   ,       ,       101,    ,       +
+,      DESVAR, 3,      4
DVPREL2,7,      PBEAM,  100,    -169,   ,       ,       102,    ,       +
+,      DESVAR, 3,      4
DVPREL2,8,      PBEAM,  100,    -170,   ,       ,       103,    ,       +
+,      DESVAR, 3,      4
DVPREL2,9,      PBEAM,  100,    -175,   ,       ,       104,    ,       +
+,      DESVAR, 4
DVPREL2,10,     PBEAM,  100,    -177,   ,       ,       105,    ,       +
+,      DESVAR, 4
$
$...FIRST INTERMEDIATE STATION (COPY OF END B DATA): (A,I1,I2,C2,D2)
$
$DVPREL2,ID,    TYPE,   PID,    FID,    PMIN,   PMAX,   EQID,   ,       +
$+,     DESVAR, DVID1,  DVID2,  ...,    ,       ,       ,       ,       +
$+,     DTABLE, CID1,   CID2,   ...
DVPREL2,11,     PBEAM,  100,    -24,    ,       ,       101,    ,       +
+,      DESVAR, 3,      4
DVPREL2,12,     PBEAM,  100,    -25,    ,       ,       102,    ,       +
+,      DESVAR, 3,      4
DVPREL2,13,     PBEAM,  100,    -26,    ,       ,       103,    ,       +
+,      DESVAR, 3,      4
DVPREL2,14,     PBEAM,  100,    -31,    ,       ,       104,    ,       +
+,      DESVAR, 4
DVPREL2,15,     PBEAM,  100,    -33,    ,       ,       105,    ,       +
+,      DESVAR, 4
$
DEQATN  101     AREA(B,H)  =  B*H
DEQATN  102       I1(B,H)  =  H*B**3/12.
DEQATN  103       I2(B,H)  =  B*H**3/12.
DEQATN  104       C2(H)    =  H/2.
DEQATN  105       D2(H)    = -H/2.
$
$DRESP1,ID,     LABEL,  RTYPE,  PTYPE,  REGION, ATTA,   ATTB,   ATT1,   +
$+,     ATT2,   ...
DRESP1, 7,      D2,     DISP,   ,       ,       3,      ,       2
DRESP1, 8,      W,      WEIGHT
$
$DCONSTR,DCID,  RID,    LALLOW, UALLOW
DCONSTR, 10,    7,      -3.0,   3.0
$
$.......2.......3.......4.......5.......6.......7.......8.......9.......0
ENDDATA
```

Listing 2-1.

2.5 Relating Design Variables to Shape Changes

To use shape sensitivity and optimization in NX Nastran, you must define design variables, and relate them to allowable shape variations. The amount the design variable is changed during optimization results in a corresponding shape change.

The allowable shapes are defined using shape basis vectors. The engineer uses these to describe how the structure is allowed to change. The optimizer then determines how much the structure can change by modifying the design variables. There are four methods are available to describe these shape basis vectors:

- Manual grid variation.
- Direct input of shapes.

- Geometric boundary shapes.
- Analytic boundary shapes.

This section begins with a brief theoretical discussion of shape basis vectors, followed by an introduction to auxiliary models, which you can use in NX Nastran as an aid to shape basis vector generation. Following these discussions is an overview of the various modeling methods available as well as an example highlighting some of the differences among these various approaches.

Other examples can be found in Example Problems, Example Problems. Interested readers may also refer to Example Problems for details regarding shape sensitivity analysis.

Basis Vectors in Shape Optimization

Shape basis vectors are used in NX Nastran to describe the properties of characteristic, allowable shape changes. The optimizer then determines the best linear combination of these vectors, given the design criteria established by the engineer.

Although the mathematics of shape basis vectors are quite simple, it is probably best to introduce the idea with a simple example.

The simple angle bracket redesign of Figure 2-9 can be used to illustrate the properties of shape basis vectors.

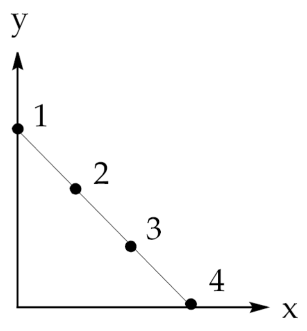


Figure 2-9. Angle Bracket

Suppose the design goal is to vary the lengths of the x- and y-axis legs of the bracket while keeping the outer edge straight. The location of all grid points along this edge can easily be described in terms of the bracket leg lengths. For simplicity, assuming only four grids along this edge, we can write

$$\begin{Bmatrix} G_{2x} \\ G_{2y} \\ G_{3x} \\ G_{3y} \end{Bmatrix} = \begin{bmatrix} 0 & 1/3 \\ 2/3 & 0 \\ 0 & 2/3 \\ 1/3 & 0 \end{bmatrix} \begin{Bmatrix} G_{1y} \\ G_{4x} \end{Bmatrix}$$

Equation 2-19.

A more realistic model certainly would include more grid points along this edge in addition to a number of grids in the structure's interior. Adding these grids would simply add more equations to the set in Eq. 2-19, leaving its basic form unchanged.

We can write Eq. 2-19 in terms of grid variations as

$$\begin{Bmatrix} \Delta G_{2x} \\ \Delta G_{2y} \\ \Delta G_{3x} \\ \Delta G_{3y} \end{Bmatrix} = \begin{bmatrix} 0 & 1/3 \\ 2/3 & 0 \\ 0 & 2/3 \\ 1/3 & 0 \end{bmatrix} \begin{Bmatrix} \Delta G_{1y} \\ \Delta G_{4x} \end{Bmatrix}$$

Equation 2-20.

The coefficient matrix is unchanged in Eq. 2-20 from that of Eq. 2-19.

Eq. 2-20 suggests that a convenient choice of design variables would be

$$\begin{aligned} \Delta x_1 &= \Delta G_{1y} \\ \Delta x_2 &= \Delta G_{4x} \end{aligned}$$

Equation 2-21.

$$\text{or } \begin{Bmatrix} \Delta G_{1y} \\ \Delta G_{2x} \\ \Delta G_{2y} \\ \Delta G_{3x} \\ \Delta G_{3y} \\ \Delta G_{4x} \end{Bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 1/3 \\ 2/3 & 0 \\ 0 & 2/3 \\ 1/3 & 0 \\ 0 & 1 \end{bmatrix} \begin{Bmatrix} \Delta x_1 \\ \Delta x_2 \end{Bmatrix}$$

Equation 2-22.

Note that Eq. 2-22 expresses a vector of grid coordinate changes in terms of a vector of design variable changes. Each column of the matrix on the right-hand side of the equation is called a shape basis vector. Generating the shape basis vectors is one of the design engineer's primary tasks in shape optimization.

If we graph each column of the matrix in Eq. 2-22 as a set of grid displacements, we see that column 1 represents a vertical (or y-component) variation in shape, while column 2 represents a horizontal shape variation. (The column order is arbitrary and just depends on the order of the design variables.)

Note that both shape basis vectors, as well as any linear combination of them, keep the edge of the bracket as a straight line, consistent with our design goals.

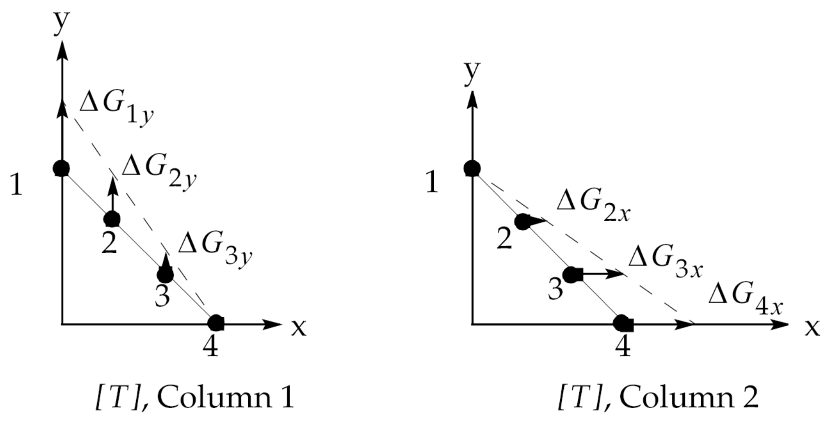


Figure 2-10. Basis Vectors for the Angle Bracket

Once a shape basis vector description is available, the optimizer can vary the shape of the structure by changing the corresponding design variables.

Reduced Basis Formulations

We can generalize the relation given in Eq. 2-22 as

$$\{\Delta G\}_{mx1} = [T]_{mxn}\{\Delta X\}_{nx1}$$

Equation 2-23.

where:

$$\{\Delta G\} = \{G\}^{i+1} - \{G\}^{i}$$
$$\{\Delta X\} = \{X\}^{i+1} - \{X\}^{i}$$

Equation 2-24.

and *i* = design cycle number.

Eq. 2-23 expresses a vector of grid point changes as a function of changes in design variables. The linear combination of design variable changes times each shape basis vector results in the total change in shape. Typically, there are more grid points than design variables, or *m* >>*n*. For this reason, Eq. 2-23 is often called a reduced basis formulation.

Auxiliary Models in Shape Optimization

An auxiliary model is simply an additional finite element model used to generate shape basis vectors. Rather than tediously specifying shape changes on an individual grid-by-grid basis, auxiliary models are a tool that can be used to simplify this process. An auxiliary model usually shares the same geometry, element connectivity, and (possibly) material type as the original structure. However, boundary conditions usually differ. Applying loads to this structure will result in sets of displacement vectors $\{U\}$. When we think of these displacement components as representing individual components of grid motion, we realize that the $\{U\}$ vectors can be quite conveniently used as basis vectors for shape optimization.

For example, consider the cantilever beam of Figure 2-11. The analysis model is clamped along the left-hand edge and tip-loaded on the right. Suppose we would like to redesign the shape of the structure by modifying the profile of the lower edge. In order to do so, we might like to use an auxiliary model to generate a few characteristic profiles and use these as our shape basis vectors.

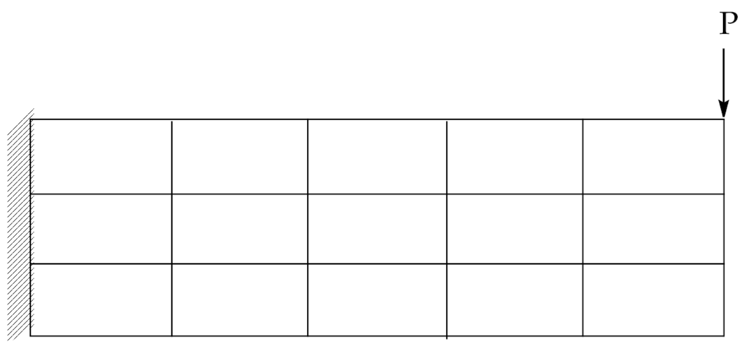


Figure 2-11. Simple 2-D Cantilever

Figure 2-12 shows the corresponding auxiliary model and its boundary conditions, exclusive of the loading. The fixed edges A and B guarantee that the left-hand support and the horizontal upper edge will not change during shape optimization. The roller supports along edge C will allow the depth of the beam to change without changing its length. We can now load edge D any way we want to produce a range of "shapes" or basis vectors.

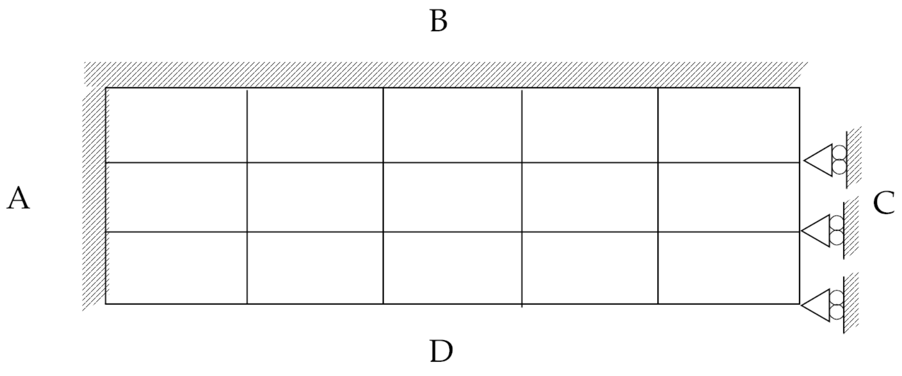


Figure 2-12. Auxiliary Model for the Simple 2-D Cantilever

A uniform taper component might be generated by applying SPCDs as shown in Figure 2-13. Here, the displacements have been plotted to show the basis vector characteristics. Visualization of these basis vectors prior to their use in connection with shape optimization is highly recommended. (An alternate approach might be to add some very stiff beams to the bottom edge of the auxiliary model. Loading the tip of the rightmost beam would then approximate the effect of the SPCD's used here.)

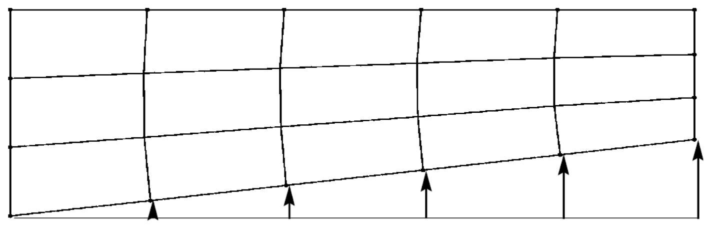


Figure 2-13. Uniform Taper Basis Vector

A more arbitrary shape basis vector could be generated by loading one of the individual grids along the lower edge. The produces an indent as shown in Figure 2-14. A number of these shapes produced by loading other grids along this edge might comprise a useful set of shape basis vectors, depending on the desired effect.

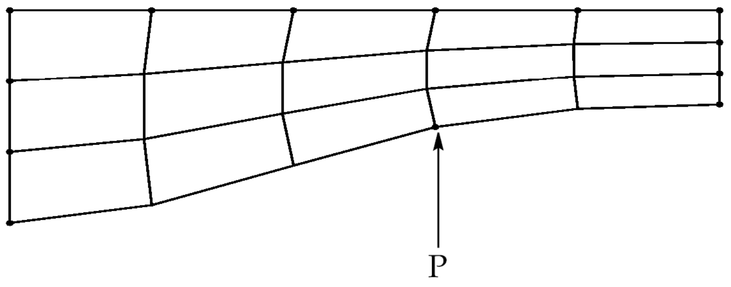


Figure 2-14. Point-Load Basis Vector

Modeling Methods (Shape Basis Vector Definition)

Basic Assumptions

Shape optimization in NX Nastran assumes the engineer is starting with a reasonably good design but would like to investigate ways in which the shape of the part, or even the entire structure, might be modified in order to better meet the design goals.

Although large shape variations can be investigated in NX Nastran, it is important to note that in a practical design environment, shape changes are often limited by numerous design considerations. Manufacturability of the part, interference from neighboring components of the structure, aesthetics, and so on, often limit the degree to which the structural shape can be modified.

Shape optimization in NX Nastran allows you to investigate these “real world” types of shape variations. It will, for example, let you determine the optimum placement and size of cutouts to lighten a structure. What the code will not tell you is how many of these cutouts to start with or whether an alternate method of construction is better. In fact, the latter questions are still active areas of shape optimization research.

Goal of Design Modeling for Shape

Four different modeling methods are available in NX Nastran. The purpose of each method is the same—to define a basis vector (or a set of basis vectors) for shape optimization. The methods only differ in terms of their user interfaces and whether or not the basis vectors are automatically updated on each design cycle. The following subsections briefly outline each approach, its strong as well as its possible weak points, and its requirements for use.

Manual Grid Variation

This is the most general, as well as perhaps the most tedious, method of generating Shape basis vectors. It is listed first since it can be thought of as the lowest level approach, somewhat analogous to Assembly Language versus FORTRAN programming. A DVGRID Bulk Data entry defines the direction and magnitude of a grid variation for a given change in a design variable. This relation is shown in Figure 2-15 and Eq. 2-25. A single DVGRID entry is required for every design variable-grid pair.

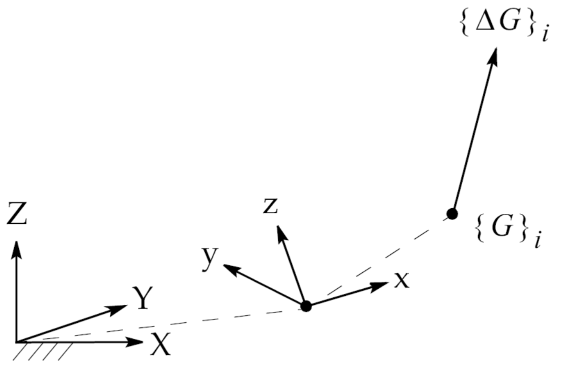


Figure 2-15. Grid Point Variation Defined by a DVGRID Entry

$$\begin{Bmatrix} \Delta G_x \\ \Delta G_y \\ \Delta G_z \end{Bmatrix}_i = \left(COEFF \begin{Bmatrix} N1 \\ N2 \\ N3 \end{Bmatrix} \right)_{ij} \Delta x_j$$

Equation 2-25.

DVGRID entries alone can be used to describe shape changes, although the method is probably more useful when combined with the direct input of shapes approach. Its most powerful form, however, is in connection with the geometric shapes approach. (These methods are described later in this section.)

Benefits

Since this can be considered the lowest-level approach, its strength lies in its generality. Using DVGRIDs alone, the designer has direct control over every designed grid point in the model (that is, every grid point whose location is to change during shape optimization.)

Drawbacks

In all but the simplest of problems, the data input can be formidable without a preprocessor. The resultant basis vectors are treated as constant and not updated with each design cycle, so the risks of mesh distortion are increased.

Checklist

1. Define the shape design variables using DESVAR Bulk Data entries.

2. Establish the corresponding grid variations using DVGRID Bulk Data entries, one entry for every design variable-designed grid pair in the model. This entry ties design variable changes to changes in the structure's shape. (Of course, this may lead to a large amount of data, up to G*NDV where G is the number of designed grids in the model and NDV is the number of shape design variables. A model with only 100 grids and 2 design variables would thus require on the order of 200 DVGRID entries.)

3. Preview the resultant basis vectors, and modify if necessary.

Direct Input of Shapes

This approach greatly simplifies the process of defining shape basis vectors. With this method, externally-generated vectors are DBLOCATEd and used to define shape basis vectors. An auxiliary model analysis provides these externally-generated vectors.

Benefits

Basis vectors for shape optimization can be generated using an external auxiliary model analysis and DBLOCATEd, allowing for easy generation of shape basis vectors. In theory, any method of external generation is possible, as long as the number of degrees of freedom in the auxiliary model is the same as in the primary structure (i.e., G-sets must be equivalent).

Drawbacks

The process is not fully automatic since an auxiliary model analysis must be performed beforehand, saved on the database, and DBLOCATEd for shape optimization. Since the basis vectors are externally generated, they are not updated for each design cycle. This may cause mesh distortion problems for large shape changes.

Checklist

1. Define an auxiliary model, perform an analysis, and save the results to the database using the SCR=NO option on the NASTRAN job submittal command. This process is the same as a conventional NX Nastran analysis except that the auxiliary model geometry and boundary conditions have been established with consideration given to the shape redesign goals.

2. DBLOCATE the results for shape optimization using the following FMS section commands (assuming the master file is file1.MASTER):

```
ASSIGN FILE1 = “file1.MASTER”
DBLOCATE DATABLK=(UG/UGD,GEOM1/GEOM1D,GEOM2/GEOM2D),
LOGICAL=FILE1
```

Note: The data blocks UG, GEOM1, and GEOM2, are DBLOCATEd and renamed to UGD, GEOM1D, and GEOM2D, respectively. The data block names UGD, GEOM1D, and GEOM2D cannot be changed.

3. Define shape design variables using DESVAR entries, and correlate these to the DBLOCATEd basis vectors using DVSHAP Bulk Data entries.

4. Preview the resultant shape basis vectors to check for modeling errors.

Geometric Boundary Shapes

This method defines allowable shape variations using only the boundary of the structure. These shape variations can be supplied manually or with a geometry-based preprocessor. Either approach relies on BNDGRD Bulk Data entries to define the structure's boundaries and DVGRID entries to furnish the shape variations over these boundaries.

Shape basis vectors are automatically generated by the code through a process of interpolation of the boundary shape changes to the interior of the structure. Additionally, the shape basis vectors are updated on every design cycle, minimizing the problems associated with mesh distortion for large shape changes.

Benefits

Since grid variations need only be specified over the boundaries, data preparation is greatly simplified. In fact, since DVGRID entries are only written for the boundaries, many real-world problems can be effectively solved without geometry-based preprocessors. Internal computation of the shape basis vectors means that these can be recomputed (updated) for every design cycle, reducing the problems associated with mesh distortion for large shape changes.

Drawbacks

Describing grid variations over the boundaries may still require a lot of DVGRID information, even though orders of magnitude less than the manual grid variation method.

Checklist

1. Define the shape design variables using DESVAR Bulk Data entries.

2. Define corresponding shape variations over the structure's boundaries using DVGRID entries.

3. Define the shape boundary conditions using BNDGRID Bulk Data entries. The BNDGRID-DVGRID entry combination is analogous to the use of SPCDs to impose enforced boundary displacements. The DVGRID entry supplies the enforced variation, much like an SPCD, to the boundary grids specified on the BNDGRID entry (similar to an SPC entry).

Analytic Boundary Shapes

This approach is similar to the geometric boundary shapes method but avoids having to use DVGRID entries on the boundary. Instead, the designer provides an auxiliary model over the boundaries of the structure that are to be changed. This may be a collection of BAR elements along the edges of a two-dimensional structure or a "skin" of plate elements over a three-dimensional part. We often refer to this model over the boundaries as an auxiliary boundary model. The designer then constrains and statically loads the model to produce a shape variation over the boundary, which the code then interpolates to the interior of the structure. The result is a shape basis vector for optimization.

Benefits

The method is very general, and does not require a geometry-based pre- and postprocessor. The user interface is entirely within the NX Nastran environment. Shape basis vectors are updated on every design cycle, reducing the problems associated with mesh distortion for large shape changes.

Drawbacks

Some creativity is often required in the selection of loads and boundary conditions for the auxiliary boundary models. However, any combination of static loading available in NX Nastran can be used; point loads (FORCE), enforced displacements (SPCD), thermal loads (TEMP), pressure loads (PLOAD), and so on, can all be used to generate the desired shapes.

Checklist

1. Define auxiliary boundary models using one additional Bulk Data Section for each model. These sections are identified with the command, BEGIN BULK AUXMODEL = n where n is the auxiliary boundary model ID.

2. Select loading and boundary conditions in Case Control using AUXCASE and AUXMODEL = n to define the auxiliary boundary model Case Control Sections.

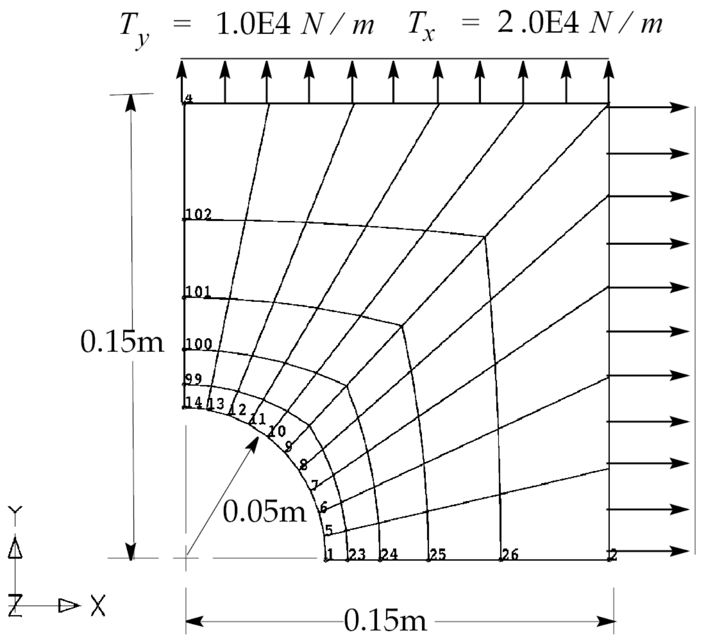
3. Define the shape boundary conditions on the actual structure using BNDGRID Bulk Data entries. (The actual structure is often referred to as the primary structure.) In addition to defining the connection points with the auxiliary boundary model, the BNDGRID entries define those boundaries that are invariant during shape optimization.

4. Define the shape design variables using DESVAR entries, and relate these to the shape basis vectors (which the code will compute) with DVSHAP entries.

Example: Shape Basis Vectors

This example illustrates the process of defining shape basis vectors for each of the previously outlined methods, excluding the manual grid variation method for which data preparation can be quite extensive. Since DVGRIDs are used in the geometric boundary shapes method anyway, their use will be covered here in connection with that method.

Suppose we want to redesign the shape of a hole cutout in a square plate shown in Figure 2-16. This redesign seeks to minimize the weight, subject to constraints on von Mises stresses. However, rather than allow only circular variations, we would like to investigate elliptical shape changes. Since the edge traction force T_x is twice T_y , one would expect a 2:1 ratio elliptical hole at the optimum, rather than simply a circle. Two design variables will be used—each one representing an axis length.



Material:
aluminum, 7075-T6 sheet

$$E = 7.2E10\ N/m^2$$
$$\mu = 0.33$$
$$\rho = 2.8E3\ kg/m^3$$

Figure 2-16. Plate Quarter Model

Recall that the equation for an ellipse centered at the origin of the x-y plane is given by

$$\frac{x^2}{a^b} + \frac{y^2}{b^2} = 1$$

Equation 2-26.

where *a* and *b* are the *x* and *y* axis intercepts, respectively, of the ellipse. We want the optimizer to vary the shape of the ellipse, so a natural choice is to define two design variables, each representing the changes in *a* and *b*. These quantities can be defined as

```
$ define design variables...
$DESVAR,ID,     LABEL,  XINIT,  XLB,    XUB,    DELXV
DESVAR, 1,      DELA,   1.0,    -20.,   +20.
DESVAR, 2,      DELB,   1.0,    -20.,   +20.
```

These two entries define two design variables: each with initial values of 1.0, lower bounds of -20., and upper bounds of +20. The initial design variable values are somewhat arbitrary as are the lower and upper bounds. (The lower bound must be less than XINIT, which in turn must be less than the upper bound, or a fatal message will result.)

Note from Eq. 2-23 that a change in a design variables value causes a change in grid location. The characteristics of the shape changes depend on the shape basis vectors. DESVAR entries 1 and 2 just define the presence of basis vector multipliers that the optimizer is free to vary. The bounds on the design variables given here allow up to a 2000% change. The basis vector magnitudes will govern the corresponding magnitude of shape change.

Direct Input of Shapes

Direct input of shapes uses an external auxiliary model to generate shape basis vectors. The auxiliary model is used to generate a set of displacement vectors that are then DBLOCATEd and identified as shape basis vectors in the optimization run. All data for the auxiliary model analysis must be provided for by the designer in a prior NX Nastran run.

Figure 2-17 shows the auxiliary model and its boundary conditions. Its geometry and connectivity are the same as for the primary structure. The material types are also the same, although this need not be the case. Note that the chosen boundary conditions are consistent with our shape redesign goals: outer edges fixed, symmetry along the x and y axes, and a free edge along the hole boundary.

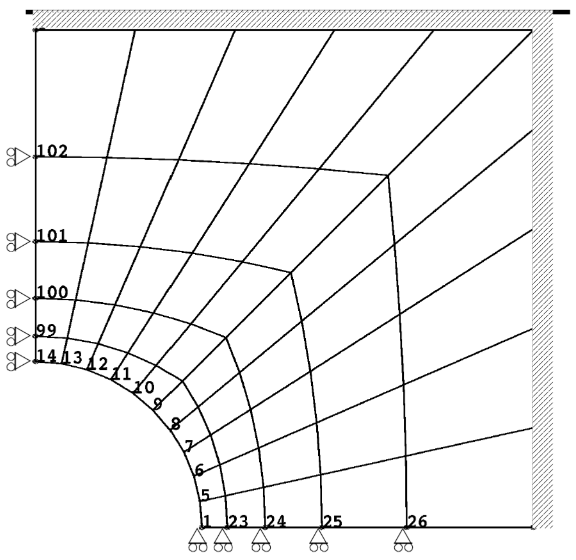


Figure 2-17. External Auxiliary Model

Along the cutout, we will apply two sets of enforced displacements: one an elliptical variation in the a, or x, direction, the other an elliptical variation in the b, or y, direction. We can use a geometry-based preprocessor to give us this data or, for simple cases, we can just compute this data by hand. Either way, the following SPCD entries result:

```
$
$ "LOAD" 1: ELLIPTICAL X-COMPONENT VARIATION (10MM MAX. VALUE)
SPCD,   101,    1,      1,      .01
SPCD,   101,    5,      1,      .0098769
SPCD,   101,    6,      1,      .0095106
SPCD,   101,    7,      1,      .0089101
SPCD,   101,    8,      1,      .0080902
SPCD,   101,    9,      1,      .0070711
SPCD,   101,    10,     1,      .0058779
SPCD,   101,    11,     1,      .0045399
SPCD,   101,    12,     1,      .0030902
SPCD,   101,    13,     1,      .0015644
SPCD,   101,    14,     1,      0.0
SPC1,   102,    123456, 1,      5,      6,      7,      8,      9,      +
+,      10,     11,     12,     13,     14
SPCADD, 103,    1,      102
$
$
$ "LOAD 2": ELLIPTICAL Y-COMPONENT VARIATION (10MM MAX. VALUE)
SPCD,   104,    5,      2,      .0015644
SPCD,   104,    6,      2,      .0030902
SPCD,   104,    7,      2,      .0045399
SPCD,   104,    8,      2,      .0058779
SPCD,   104,    9,      2,      .0070711
SPCD,   104,    10,     2,      .0080902
SPCD,   104,    11,     2,      .0089101
SPCD,   104,    12,     2,      .0095106
SPCD,   104,    13,     2,      .0098769
SPCD,   104,    14,     2,      .01
SPC1,   105,    123456, 1,      5,      6,      7,      8,      9,      +
+,      10,     11,     12,     13,     14
SPCADD, 106,    1,      105
$
```

Note that the SPCD enforced displacement values correspond to an elliptical variation along the hole boundary with a maximum value of 10 mm. The result of these load applications are shown in Figure 2-18 and Figure 2-19. (The deformations have been enlarged for clarity.) These displacement vectors are used next to generate our shape basis vectors.

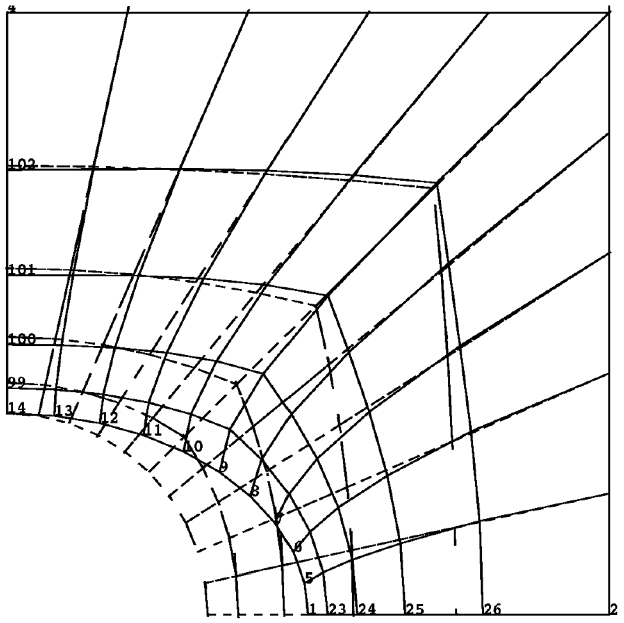


Figure 2-18. Displacement Vector 1

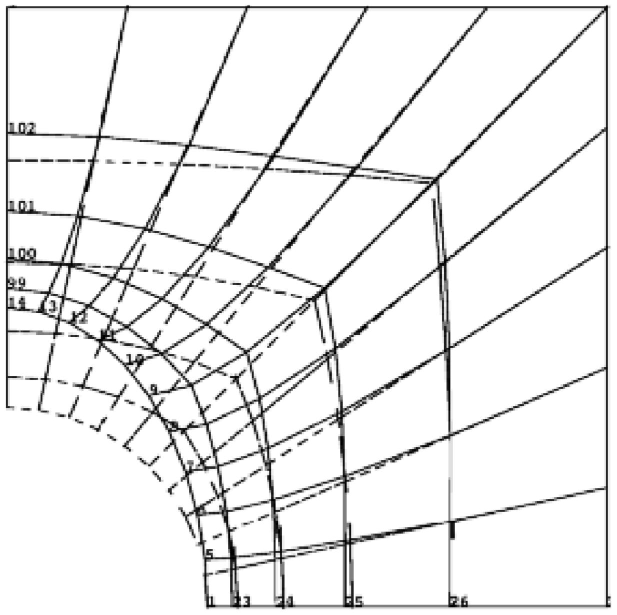


Figure 2-19. Displacement Vector 2

In the optimization run, we will DBLOCATE these displacement vectors and use this data to define our shape basis vectors. This data can be DBLOCATEd using the following FMS Section statements:

```
assign f1 = 'plate33_aux.MASTER'
dblocate datablk=(ug/ugd,geom1/geom1d,geom2/geom2d), logical=f1
```

where plate33_aux.MASTER is the master DBset file corresponding to the auxiliary model analysis results shown in Figure 2-18 and Figure 2-19.

Having input this information, we need to identify the design variables that are to act as multipliers of these basis vectors. This is done using the following DVSHAP Bulk Data entries:

```
$DVSHAP,DVID,   COL1,   SF1,    COL2,   SF2,    ...
DVSHAP, 1,      1,      1.0
DVSHAP, 2,      2,      1.0
```

The first entry defines Design Variable 1 (DVID = 1) as the multiplier of the first DBLOCATEd displacement vector (COL1 = 1) using the factor 1.0 (SF1) as the constant of multiplication. This process defines the first shape basis vector. A similar process on the second DVSHAP entry defines the next shape basis vector.

Note that we now have two basis vectors, one for each of the *a* and *b* elliptical variations of the hole boundary. As the optimizer changes x1 and x2, the grid locations are varied according to

$$\{\Delta G\} = \begin{bmatrix} 1.0u_1 & 1.0u_2 \end{bmatrix} \begin{Bmatrix} \Delta x_1 \\ \Delta x_2 \end{Bmatrix}$$

Equation 2-27.

where u_1 is the first DBLOCATEd basis vector (note the 1.0 multiplying factor) and u_2 is the second.

Geometric Boundary Shapes

Rather than externally generating the shape basis vector data as in the previous approach, this method generates the vectors automatically, based on geometric data supplied on the boundaries. All of the data preparation and input appear in the optimization input file; there is no need for a prior auxiliary model analysis.

In addition, the shape basis vectors are updated on each design cycle as the shape changes. One problem with basis vectors that are not updated is that they may lead to ill-conditioned meshes for large shape changes. Updating this data on each design cycle minimizes (but not altogether eliminates) this difficulty.

The geometric boundary shapes method begins with a definition of allowable variations over the boundaries. These variations can be described with a geometry-based preprocessor, or with Bulk Data alone. Either way, the boundaries are defined using BNDGRID entries, and the variations defined using DVGRID entries.

The elliptical variations on the boundaries can be defined as follows:

```
$ DEFINE DESIGN VARIABLES...
DESVAR, 1,      DELA,   1.0,    -20.,   +20.
DESVAR, 2,      DELB,   1.0,    -20.,   +20.
$
$ ...AND RELATE THESE TO BOUNDARY VARIATIONS DEFINED USING DVGRIDS:
$DVGRID,DVID,   GRID,   CID,    COEFF,  N1,     N2,     N3
$
$ BASIS VECTOR 1:
DVGRID, 1,      1,      ,       .01,     1.0,   0.0,    0.0
DVGRID, 1,      5,      ,       .0098769,1.0,   0.0,    0.0
DVGRID, 1,      6,      ,       .0095106,1.0,   0.0,    0.0
DVGRID, 1,      7,      ,       .0089101,1.0,   0.0,    0.0
DVGRID, 1,      8,      ,       .0080902,1.0,   0.0,    0.0
DVGRID, 1,      9,      ,       .0070711,1.0,   0.0,    0.0
DVGRID, 1,      10,     ,       .0058779,1.0,   0.0,    0.0
DVGRID, 1,      11,     ,       .0045399,1.0,   0.0,    0.0
DVGRID, 1,      12,     ,       .0030902,1.0,   0.0,    0.0
DVGRID, 1,      13,     ,       .0015644,1.0,   0.0,    0.0
DVGRID, 1,      14,     ,       .0,      1.0,   0.0,    0.0
$
$
$ BASIS VECTOR 2:
DVGRID, 2,      1,      ,       .0,      0.0,   1.0,    0.0
DVGRID, 2,      5,      ,       .0015644,0.0,   1.0,    0.0
DVGRID, 2,      6,      ,       .0030902,0.0,   1.0,    0.0
DVGRID, 2,      7,      ,       .0045399,0.0,   1.0,    0.0
DVGRID, 2,      8,      ,       .0058779,0.0,   1.0,    0.0
DVGRID, 2,      9,      ,       .0070711,0.0,   1.0,    0.0
DVGRID, 2,      10,     ,       .0080902,0.0,   1.0,    0.0
DVGRID, 2,      11,     ,       .0089101,0.0,   1.0,    0.0
DVGRID, 2,      12,     ,       .0095106,0.0,   1.0,    0.0
DVGRID, 2,      13,     ,       .0098769,0.0,   1.0,    0.0
DVGRID, 2,      14,     ,       .01,     0.0,   1.0,    0.0
```

The first set of DVGRID entries are related to Design Variable 1, DELA. Note that the coefficient magnitudes (COEFF) are the same as the SPCD magnitudes of the previous example. We observe that this set describes an elliptical shape variation in the x-direction (N1 = 1.0, N2 = N3 = 0.0), dependent on changes in Design Variable 1. The second set of DVGRID entries likewise describes an elliptical shape variation in the y-direction, which is dependent on changes in Design Variable 2.

BNDGRID entries complete the shape boundary definition process and identify two categories of grid components:

- Grid components that change according to data supplied on the DVGRID entries.
- Grid components that remain fixed.

The process is similar to the SPC, SPCD combination used to impose enforced displacements in static analysis. The DVGRID entry supplies the grid motion on the boundary, much like an SPCD enforced displacement. The boundary grids are identified using BNDGRID entries, much like an SPC entry that identifies the corresponding grids and their components.

```
$
$ BOUNDARY CONDITIONS FOR SPCD APPLICATION (PROVIDED VIA DVGRIDS)
$       OUTER PLATE EDGES...
BNDGRID,123456, 2,      15,     16,     17,     18,     3,      19,     +
+,      20,     21,     22,     4
$       X=0 PLANE...
BNDGRID,23456,  23,     24,     25,     26
$       Y=0 PLANE...
BNDGRID,13456,  99,     100,    101,    102
$       CUTOUT EDGE...
BNDGRID,123456, 1,      5,      6,      7,      8,      9,      10,     +
+,      11,     12,     13,     14
```

Note

The boundary conditions defined by the DVGRID and BNDGRID entries are only used to generate shape basis vectors. They are not used in connection with the primary model analysis.

The first BNDGRID entry identifies all grid components along the plate outer edges. Since no DVGRIDs have been furnished for these degrees of freedom, their displacements will be fixed to zero. For shape optimization, the plate outer edges will be invariant.

The second BNDGRID entry identifies components 2 through 6 for those grids that lie on the x-axis. This allows the x-component to vary during shape optimization, consistent with the elliptical changes made to the cutout. All other components are considered fixed since no enforced shape changes for these degrees of freedom have been supplied on DVGRID entries. The third BNDGRID entry furnishes similar information as the previous entry, only with regard to shape changes along the y-axis.

The last BNDGRID entry identifies components 1 through 6 for the cutout edge grids. Shape changes are enforced for components 1 and 2 since they have been provided on DVGRID entries. All other components are considered fixed.

Once the free, fixed, and enforced portions of the shape changes have been defined, the code interpolates this information to all of the interior degrees of freedom. In this case, the resulting shape basis vectors are identical to those shown in Figure 2-18 and Figure 2-19.

On subsequent iterations, the shape basis vectors may change slightly because the interpolated solutions over the interior of the structure will change due to the modified geometry. Recomputing the basis vectors at the beginning of each design cycle helps to minimize the problems associated with mesh distortion.

Analytic Boundary Shapes

This method also uses auxiliary models. Here, however, they are defined over the boundaries of the structure only. The so-called auxiliary boundary models can be an edge of BAR elements for a two-dimensional redesign or a skin of shell elements for a general, three-dimensional shape optimization task.

Auxiliary boundary models are used to generate shape variations over the boundaries of the structure, providing a shortcut to the geometric boundary shapes method. You can define as many auxiliary boundary models as are necessary using any number and types of applied static loading to produce sets of desired boundary variations. The code interpolates these boundary changes to the structure's interior, thus automatically generating the shape basis vectors.

Each auxiliary boundary model requires its own Bulk Data Section. Each model can be loaded in any number of subcases to produce more than one shape basis vector per auxiliary boundary model. Load and boundary condition sets are selected using the auxiliary boundary model Case Control Sections.

The following Bulk Data Section provides the auxiliary boundary model definition. This section must appear after the Bulk Data for the primary model.

```
BEGIN BULK AUXMODEL = 1
$
PARAM, PRGPST, NO
$
$ auxiliary model element definition (geometry from primary structure):
$
CBAR,   101,    20,     1,      5,      0.,     0.,     1.
CBAR,   102,    20,     5,      6,      0.,     0.,     1.
CBAR,   103,    20,     6,      7,      0.,     0.,     1.
CBAR,   104,    20,     7,      8,      0.,     0.,     1.
CBAR,   105,    20,     8,      9,      0.,     0.,     1.
CBAR,   106,    20,     9,      10,     0.,     0.,     1.
CBAR,   107,    20,     10,     11,     0.,     0.,     1.
CBAR,   108,    20,     11,     12,     0.,     0.,     1.
CBAR,   109,    20,     12,     13,     0.,     0.,     1.
CBAR,   110,    20,     13,     14,     0.,     0.,     1.
PBAR,   20,     102,    1.0E-3, 1.0E-12,1.0E-12,1.0E-12
MAT1,   102,    7.2+10, ,       0.33,   2.8015+3,1.0-2
$
$ boundary condition set I:
PLOAD1, 160,    101,    FX,     FR,     0.,     100.,   1.,     100.
PLOAD1, 160,    102,    FX,     FR,     0.,     100.,   1.,     100.
PLOAD1, 160,    103,    FX,     FR,     0.,     100.,   1.,     100.
PLOAD1, 160,    104,    FX,     FR,     0.,     100.,   1.,     100.
PLOAD1, 160,    105,    FX,     FR,     0.,     100.,   1.,     100.
PLOAD1, 160,    106,    FX,     FR,     0.,     100.,   1.,     100.
PLOAD1, 160,    107,    FX,     FR,     0.,     100.,   1.,     100.
PLOAD1, 160,    108,    FX,     FR,     0.,     100.,   1.,     100.
PLOAD1, 160,    109,    FX,     FR,     0.,     100.,   1.,     100.
PLOAD1, 160,    110,    FX,     FR,     0.,     100.,   1.,     100.
$
$ boundary condition set II:
PLOAD1, 161,    101,    FY,     FR,     0.,     100.,   1.,     100.
PLOAD1, 161,    102,    FY,     FR,     0.,     100.,   1.,     100.
PLOAD1, 161,    103,    FY,     FR,     0.,     100.,   1.,     100.
PLOAD1, 161,    104,    FY,     FR,     0.,     100.,   1.,     100.
PLOAD1, 161,    105,    FY,     FR,     0.,     100.,   1.,     100.
PLOAD1, 161,    106,    FY,     FR,     0.,     100.,   1.,     100.
PLOAD1, 161,    107,    FY,     FR,     0.,     100.,   1.,     100.
PLOAD1, 161,    108,    FY,     FR,     0.,     100.,   1.,     100.
PLOAD1, 161,    109,    FY,     FR,     0.,     100.,   1.,     100.
PLOAD1, 161,    110,    FY,     FR,     0.,     100.,   1.,     100.
$
$ temperature distribution:
TEMP,   162,    1,      1.,     5,      1.,     6,      1.
TEMP,   162,    7,      1.,     8,      1.,     9,      1.
TEMP,   162,    10,     1.,     11,     1.,     12,     1.
TEMP,   162,    13,     1.,     14,     1.
$
$ spc conditions for both load sets:
SPC,    100,    1,      23456,  0.0
SPC,    100,    14,     13456,  0.0
SPC1,   100,    345,    5,      6,      7,      8,      9,      10,     +
+,      11,     12,     13
$
ENDDATA
```

A number of interesting things can be seen in this listing. First, the CBAR entries define the auxiliary boundary model elements: a ring of bar elements along the cutout. Note that GRID entries do not appear because grid data is shared with the primary structure, and additional grids need not be defined. (Of course, if extra grids are necessary, they can be added as required. If the location of the hole were to change, one might want to define another grid at the center to use as an attachment point for rigid elements.)

Two sets of loads are applied. One imposes an elliptical-type variation in the x-direction, and the other applies a similar displacement in the y-direction. Uniform loading with PLOAD1’s and a uniform temperature load have been used to generate these boundary deformations.

The loading and boundary condition sets are selected in the auxiliary boundary model Case Control Section, which follows the Case Control for the primary model:

```
$ AUXILIARY MODEL CASE CONTROL:
AUXCASE
   AUXMODEL = 1
   SUBCASE 10
      SUBTITLE = AUXILIARY MODEL 1, LOAD CASE 10
      SPC  = 100
      LOAD = 160
      TEMP(LOAD) = 162
      DISPLACEMENT = ALL
   SUBCASE 20
      SUBTITLE = AUXILIARY MODEL 1, LOAD CASE 20
      SPC  = 100
      LOAD = 161
      TEMP(LOAD) = 162
      DISPLACEMENT = ALL
BEGIN BULK
```

The keyword AUXCASE identifies the beginning of the auxiliary model Case Control sections. In this example, a single auxiliary boundary model (AUXMODEL = 1) is used to generate two basis vectors in subcases 10 and 20. The resultant boundary displacements are shown in Figure 2-20 and Figure 2-21.

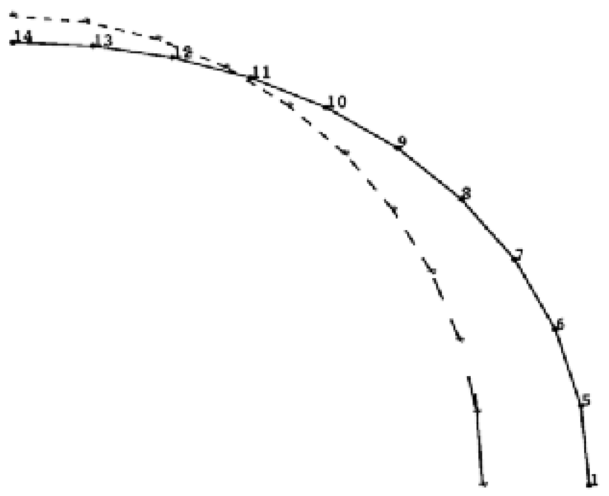


Figure 2-20. Boundary Displacement Set 1

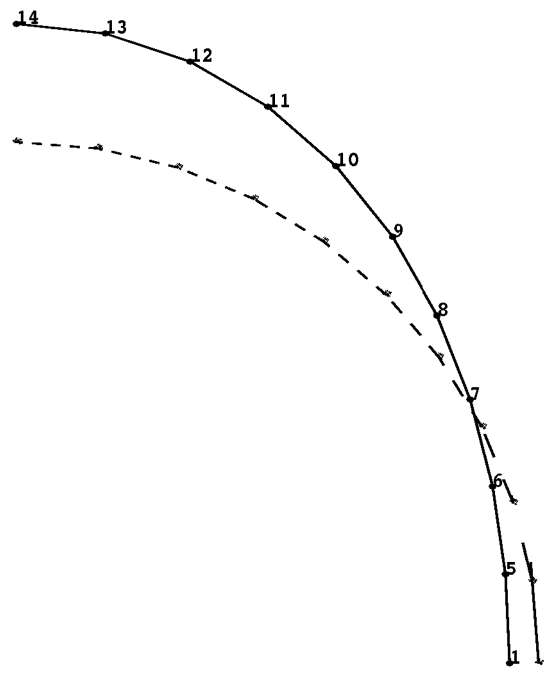


Figure 2-21. Boundary Displacement Set 2

Once these boundary displacements have been computed, the code interpolates these displacements to the interior grids. Once again, BNDGRID entries define the degrees of freedom along the boundaries that are allowed to change and those degrees of freedom that are to remain fixed.

```
$
$ boundary conditions for basis vector generation:
$       outer plate edges...
BNDGRID,123456, 2,      15,     16,     17,     18,     3,      19,     +
+,      20,     21,     22,     4
$       x=0 plane...
BNDGRID,23456,  23,     24,     25,     26
$       y=0 plane...
BNDGRID,13456,  99,     100,    101,    102
$       cutout edge...
BNDGRID,123456, 1,      5,      6,      7,      8,      9,      10,     +
+,      11,     12,     13,     14
```

You will recognize these as the same set of boundary conditions that were used in the geometric shapes method. The auxiliary boundary model solutions, combined with the BNDGRID boundary data, supply sufficient information to the code to use in interpolating these shape changes to the interior of the structure. The result is a family of shape basis vectors. Finally, the designer ties these basis vectors to changes in the set of design variables using the DVBSHAP entries:

```
$DVBSHAP,DVID,  AUXMID, COL1,   SF1,    ...
DVBSHAP,1,      1,      1,      1.0
DVBSHAP,2,      1,      2,      1.0
```

Each entry associates a Design Variable with a basis vector resulting from a particular auxiliary boundary model (AUXMID). The column number (COL) indicates the auxiliary model displacement solution number. In this example, design variable 1 is the multiplier (with a 1.0 multiple) of the shape basis vector resulting from a solution of auxiliary boundary model 1 (AUXMID = 1). But recall that this model has two subcases: 10 and 20. The COL field on the entry reconciles this. COL1 = 1 implies the first solution from subcase 10. COL1 = 2 implies the second solution from subcase 20. Similarly, design variable 2 is the second basis vector multiplier computed from the second solution (subcase 20) of auxiliary boundary model 1 (AUXMID = 1).

The resultant reduced basis formulation could be written as:

$$\{\Delta G\} = \begin{bmatrix} 1.0U_{\substack{AUXMID = 1 \\ SUBCASE = 10}} & 1.0U_{\substack{AUXMID = 1 \\ SUBCASE = 20}} \end{bmatrix} \begin{Bmatrix} \Delta x_1 \\ \Delta x_2 \end{Bmatrix}$$

Equation 2-28.

where each column is a displacement solution of the auxiliary model. Each column is the same size as $\{\Delta G\}$. The subscripts on displacement solutions indicate the source of the applied boundary displacements. The first shape basis vector is derived from the first auxiliary boundary model subcase, while the second shape basis vector is derived from the second auxiliary model subcase.

2.6 Identifying the Design Responses

Before the objective function and constraints can be defined, the analysis responses on which they depend must be identified. These responses are called design responses and are specified in the design model using DRESP1, DRESP2, and DRESP3 Bulk Data entries.

Type 1 Responses

DRESP1 Bulk Data entries define type-1, or first-level responses. These responses are available directly from an NX Nastran analysis. Structural weight, displacements at grid points, element stresses, and so on are all examples of type-1 responses. For a list of available responses, see the DRESP1 Bulk Data entry description. For information on how to generate multiple DRESP1 Bulk Data entries for more than one entity (such as a grid or an element) or frequency, and how to generate results of simple mathematical functions (such as Sum or Average) in frequency response, see Efficiencies in First-Level Response Identification.

Type 2 Responses

DRESP2 and DRESP3 Bulk Data entries define type-2, or second-level responses. These responses are also referred to as user-defined responses because the user utilizes either the built-in mathematical operations in NX Nastran or an external program to define design responses from combinations of first-level responses, design variables, grid coordinate values, and table constants.

- DRESP2 Bulk Data entries reference mathematical operations that are built into NX Nastran. A few built-in mathematical operations are available through the FUNC field of a DRESP2 Bulk Data entry. A more comprehensive library of built-in mathematical operations is available through the EQID field of a DRESP2 Bulk Data entry. The EQID field references a DEQATN (Design EQuATioN) Bulk Data entry. With a DEQATN Bulk Data entry, the user can define a response using algebraic, trigonometric, hyperbolic, and logarithmic functions. For example, the user can define design responses like error functions and stress averages.

- DRESP3 Bulk Data entries reference external programs. With external programs, the user might define, for example, design responses where:
  - o The evaluation is not direct, but may require an iterative solution (as in Newton's method) to arrive at the value for the design response.
  - o The evaluation requires the solution of a system of equations to arrive at the value for the design response.

During the optimization run, NX Nastran passes data defined in a DRESP3 Bulk Data entry to the external program. The data is passed in one-dimensional arrays. The external program receives the data and uses it to calculate a value for the design response. The design response value is then passed back to NX Nastran for use as the value of the objective function or for use in some constraints. Figure 2-22 shows the relationship between DRESP1 and DRESP3 design responses.

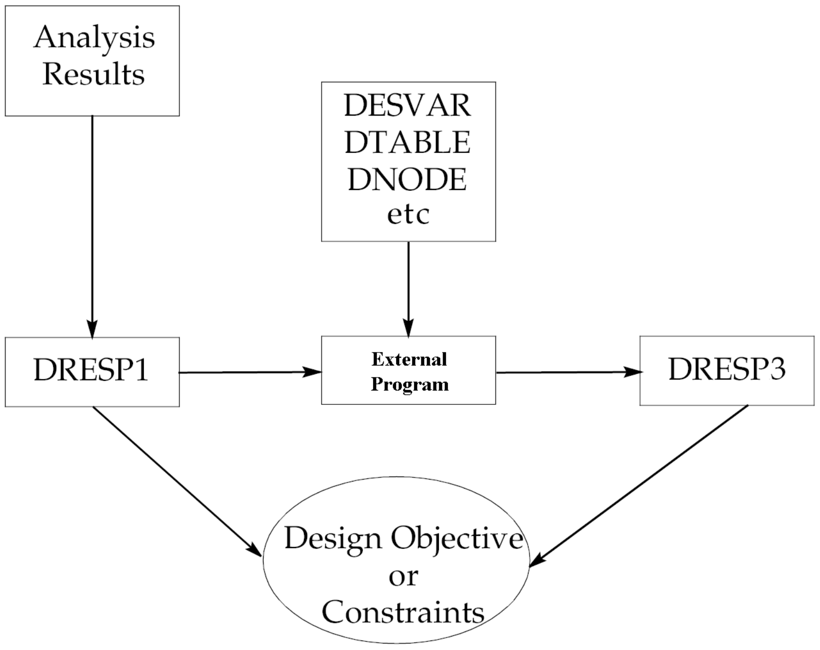


Figure 2-22. First- and Second-Level Responses

For additional information on using the DRESP3 Bulk Data entry capability, see Defining Design Responses with External Programs.

To illustrate the difference between DRESP1 and DRESP2 entries, suppose we have a case where we want to use the x and y static displacement components at a particular grid point as design responses in our design model. These first-level responses can be identified using two DRESP1 entries as follows:

```
$DRESP1,ID,     LABEL,  RTYPE,  PTYPE,  REGION, ATTA,   ATTB,   ATT1,   +
$+,     ATT2,   ...
$
$... X DISPLACEMENT AT GRID 100:
DRESP1, 501,    UX100,  DISP,   ,       ,       1,      ,       100
$... Y DISPLACEMENT AT GRID 100:
DRESP1, 502,    UY100,  DISP,   ,       ,       2,      ,       100
```

DRESP1 501 selects the x-component of displacement at Grid 100, and DRESP1 502 selects the corresponding y-component. These displacements may also be used in a synthetic relation. For example, to express the total x-y plane displacement at Grid 100 as the square root of the sum of displacement squares we could use:

```
$
$... DEFINITION OF EQUATION:
$DEQATN EQUID   F() = ...
DEQATN  510     U( UX, UY ) = SQRT( UX**2 + UY**2 )
$
$... SYNTHETIC RESPONSE:
$DRESP2,ID,     LABEL,  EQID,   REGION, ,       ,       ,       ,       +
$+,     DESVAR, DVID1,  DVID2,  ...,    ,       ,       ,       ,       +
$+,     DTABLE, LABEL1, LABEL2, ...,    ,       ,       ,       ,       +
$+,     DRESP1, NR1,    NR2,    ...,    ,       ,       ,       ,       +
$+,     DNODE,  NID1,   DIR1,   NID2,   DIR2,   ...,    ,       ,       +
DRESP2, 520,    U,      510,    ,       ,       ,       ,       ,       +
+,      DRESP1, 501,    502
$
```

The equation data is supplied on the DEQATN entry, while the DRESP2 entry defines the arguments to this equation – in this case, the two first-level responses DRESP1 501 and 502. DRESP2 520 can now be used either as the objective function or as a constraint.

Efficiencies in First-Level Response Identification

1. Since a large number of responses are often used in design, the DRESP1 entry must be able to efficiently define many responses using few entries. For example, a single DRESP1 entry can identify the z-component of displacement at grid points 100, 101, and 102 as follows:

```
$DRESP1,ID,     LABEL,  RTYPE,  PTYPE,  REGION, ATTA,   ATTB,   ATT1,   +
$+,     ATT2,   ...
$
DRESP1, 100,    UZ,     DISP,   ,       ,       3,      ,       100,    +
+,      101,    102
```

This list can be extended as necessary to any number of grids.

Note

Item codes used on DRESP1 entries to select response components can be found in the *NX Nastran Quick Reference Guide*. These are also often referred to as plot codes.

Element-level responses can be selected using either element IDs (much like the list of grids in the previous example), or property IDs. By supplying a property ID on a DRESP1 entry, we identify the element-level response for every element in that property group. For example, the following selects the axial stress response for all ROD elements in property groups 150, 160, 170 and 180:

```
$DRESP1,ID,     LABEL,  RTYPE,  PTYPE,  REGION, ATTA,   ATTB,   ATT1,   +
$+,     ATT2,   ...
$
DRESP1, 250,    SIG1,   STRESS, PROD,   ,       2,      ,       150,    +
+,      160,    170,    180
```

This can generate quite a lot of design data. Figure 2-23 is a schematic diagram of this hierarchy.

If limits are subsequently placed on these responses using a DCONSTR entry (discussed in Defining the Constraints), the result will be a pair of stress constraints (one for the upper bound and one for the lower bound) for every element in each of these property groups. By using just a pair of entries, we have been able to define potentially hundreds of stress constraints.

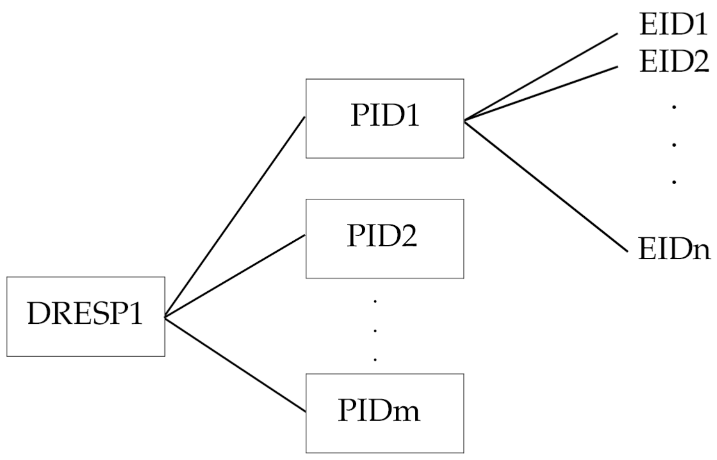


Figure 2-23. Identification of Element-Level Responses

2. Further automatic generation of DRESP1 data is possible in Frequency Response or Transient Response analyses. In a Frequency Response analysis, if you leave the frequency value field blank in the DRESP1 entry, NX Nastran generates that particular DRESP1 entry for every frequency in the relevant frequency set. If you are using an OFREQUENCY Case Control command, the software generates that particular DRESP1 frequency for all frequencies specified by the OFREQUUENCY command. Similarly, in a Transient Response analysis, if you leave the time value field blank, NX Nastran generates multiple DRESP1 entries. In addition, if you specify multiple entities (grid or element), NX Nastran generates multiple DRESP1 entries for each listed entity.

3. For Frequency Response or Transient Response analyses, you can obtain a simple mathematical function resultant of the multiple responses by entering character input in the field instead of either specifying a frequency or time value or leaving the field blank. For example, if you specify "AVG" instead of a frequency value in a DRESP1 entry, NX Nastran evaluates DRESP1 at all frequencies in the relevant set. It then calculates the average of these responses. See Remark 20 for the DRESP1 entry in the *NX Nastran Quick Reference Guide* for further details.

   Note

   If you reference such a DRESP1 entry from a DRESP2 entry, you should reference it as a DRESP2 entry. If you reference it as a DRESP1 entry, then the software generates the referencing DRESP2 entry for multiple frequencies using the frequency-based components of the mathematical function (such as of the "AVG") rather than the function resultant itself (the actual "AVG"). Such mathematical function resultants are referred to as "integrated responses" for brevity. Currently, you cannot reference these responses from a DRESP3 entry, except as a DRESP1 entry.

Efficiencies in Second-Level Response Definition

1. These efficiencies in design response identification extend to type-2 responses as well. For example, consider the following DRESP1 entries that identify major and minor principal stresses at surface z1 for PSHELL group 10:

```
$DRESP1,ID,     LABEL,  RTYPE,  PTYPE,  REGION, ATTA,   ATTB,   ATT1,   +
$+,     ATT2,   ...
$
$ Major principal stress at surface z1:
DRESP1, 101,    SIG1,   STRESS, PSHELL, ,       7,      ,       10
$
$ Minor principal stress at surface z1:
DRESP1, 102,    SIG2,   STRESS, PSHELL, ,       8,      ,       10
```

   This pair of entries defines a pair of stresses for every element in this property group. This could easily result in hundreds of design responses.

   Assume we wanted to write the maximum shearing stress as the average of these responses:

```
$DRESP2,ID,     LABEL,  EQID,   REGION, ,       ,       ,       ,       +
$+,     DESVAR, DVID1,  DVID2,  ...,    ,       ,       ,       ,       +
$+,     DTABLE, LABEL1, LABEL2, ...,    ,       ,       ,       ,       +
$+,     DRESP1, NR1,    NR2,    ...,    ,       ,       ,       ,       +
$+,     DNODE,  NID1,   DIR1,   NID2,   DIR2,   ...,    ,       ,       +
$
$ Input to equation to compute max shears:
DRESP2, 201,    MAXS,   300,    ,       ,       ,       ,       ,       +
+,      DRESP1, 101,    102
$
$ Equation for max shears:
DEQATN  300     MAXS(SIG1,SIG2) = (SIG1-SIG2)/2.
```

   This single DRESP2 entry defines a maximum shear for every element in the group. Because the underlying first-level responses are element-level, the DRESP2 is as well. This also applies to DRESP3 type responses.

2. You can multiply nest DRESP2 entries. In other words, each DRESP2 entry can reference one or more other DRESP2 entries. However, you can only nest integrated response DRESP1 entries which are referenced as DRESP2 entries a single time (i.e., you can only reference it from a DRESP2 entry that isn't further nested). You cannot nest DRESP3 entries inside each other.

Limitations in Writing Second-Level Responses

A few restrictions apply to the formulation of second-level responses:

- Unless you use a DRSPAN Case Control command, you can only reference subcase-dependent responses from a DRESP2 or DRESP3 entry with other responses from the same subcase. If a DRESP2 or DRESP3 entry contains one or more DRESP1 entries that are assigned with a DRSPAN command to particular subcases, then all DRESP1 entries in the same DRESP2 or DRESP3 entries must be assigned with the DRSPAN command. You can also use a DRSPAN command to assign any integrated response DRESP1 entries to a specific subcase.

- When combining responses of different types (e.g. stress + strain, etc.), the responses must be scalar quantities. That is, the corresponding DRESP1 entries must define a single response only. This implies that for displacements, only a single grid may be listed on the DRESP1 entry; for element-level responses, the ELEM identifier must be used with only a single element ID; and so on.

Designed Grids

The DNODE input fields on the DRESP2 entry may be used to input the coordinates for designed grids only. A designed grid is one whose coordinates may vary during shape optimization. Mathematically, a designed grid has a nonzero coordinate component in one or more of the shape basis vectors. See the DRESP2 entry for a detailed description of designed grids.

Example

Suppose we would like to include Euler buckling constraints in the design model for some simple, pin-ended rod elements. A representative element is shown in Figure 2-24.

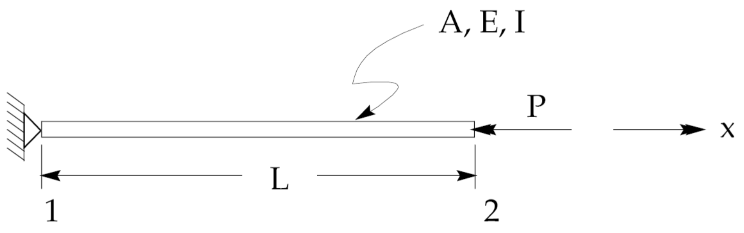


Figure 2-24. Pin-ended Rod Element

The critical load that induces the first Euler buckling mode is given by

$$P_{cr} = -\frac{\pi^2 EI}{L^2}$$

Equation 2-29.

The design requirements call for the axial load in this ROD element to be greater (less compressive) than this critical load P_{CR} or

$$\frac{P}{P_{CR}} \leq -1$$

Equation 2-30.

which can be rewritten as

$$\frac{P(x_2 - x_1)^2}{-\pi^2 EI} \leq 1$$

Equation 2-31.

Note that the critical load is a function of the least area moment of inertia, yet this quantity is not a part of the analysis model specification for the ROD element. If the cross-sectional area of the element is the design variable and a solid circular section is assumed, then

$$I = \frac{A^2}{4\pi}$$

Equation 2-32.

and

$$\frac{4P(x_2 - x_1)^2}{-\pi E A^2} \leq 1$$

Equation 2-33.

The buckling constraint is now a function of the cross-sectional area design variable for this ROD element geometry. The following Bulk Data entries illustrate one way in which these relations may be implemented:

```
--------------------------------------------------------------------------------
GRID,   1,      0,       0.,     0.,     0.
GRID,   2,      0,      10.,     0.,     0.
CROD,   100,    200,    1,      2
PROD,   200,    201,    0.5
--------------------------------------------------------------------------------
DESVAR, 1,      A200,   0.5,    0.25,   0.75
$
DVPREL1,301,    PROD,   200,    4,      ,       ,       ,       ,       +
+,      1,      1.0
$
DRESP1, 401,    SA,     FORCE,  ELEM,   ,       2,      ,       100
$
DRESP2, 501,    EUL1,   600,    ,       ,       ,       ,       ,       +
+,      DESVAR, 1,      ,       ,       ,       ,       ,       ,       +
+,      DTABLE, L,      ,       ,       ,       ,       ,       ,       +
+,      DRESP1, 401
$
DTABLE, L,      10.
$
DEQATN  600     EUL1(A,L,P) = 4.*P*L**2/(-PI(1.0)*1.0E7*A**2)
$                note use of PI function--^ (see DEQATN entry description)
$
$DCONSTR,DCID,  RID,    LALLOW, UALLOW
DCONSTR,500,    501,    -1.E35, 1.0
--------------------------------------------------------------------------------
```

The two GRID entries along with the CROD and PROD entries specify ROD Element 100 with a length of 10 units along the x-axis and an initial cross-sectional area of 0.5.

The DESVAR entry defines the area design variable with an initial value equal to the initial cross-sectional area of the ROD. Lower and upper bounds of 0.25 and 0.75, respectively, are set, representing 50% move limits on the cross-sectional area.

Since the constraint on Euler buckling is applied here for just a single element, the ‘ELEM’ identifier is used on the DRESP1 in field 5 along with a ‘100’ in field 9 to specify that the axial load for Element 100 is to be used in the design model. The force component is selected by reference to the plot codes (the 2 in field 7 indicates axial force). See the *NX Nastran Quick Reference Guide* for these plot codes.

Note

The order of the DRESP2 continuation lines is not interchangeable. See the DRESP2 Bulk Data entry description.

The DRESP2 entry defines the arguments of the Euler buckling equation DEQATN 600. Note that this information is positional; the values are assigned to the argument list of the equation based on the order of specification in the DRESP2 entry.

Defining Design Responses with External Programs

When using the DRESP3 Bulk Data entry capability for type-2 design responses, the user can:

- Customize copies of the FORTRAN subroutines that are provided in the installation directory to perform the design response calculations. A build script is also provided in the installation directory that is suitable for use with these FORTRAN subroutines.

- Modify copies of the FORTRAN subroutines that are provided in the installation directory to interface with user-coded or third-party software that performs the design response calculations.

The step-by-step procedure to customize copies of the provided FORTRAN subroutines is as follows:

1. Customize the provided FORTRAN subroutines

   Copy the contents of the *install_directory\nxnr\dr3srv* directory to another location. Then modify the FORTRAN subroutines *r3sgrt.F*, *r3svald.F* (for LP), and *r3svals.F* (for ILP) as necessary to suit your application. If necessary, add new subroutines or chains of subroutines. To include multiple responses in the same routine, separate the responses with if/else program structures and use the TYPE field on DRESP3 Bulk Data entry to specify which response to calculate.

   The data included on a DRESP3 Bulk Data entry is passed to the executable file in three one-dimensional arrays. It is necessary to code the external program so that it interprets the information contained in these arrays correctly. For details on how the arrays are formatted, see Format of Data Passed to the Executable File.

2. Create the executable file

   Once you have completed customizing the provided FORTRAN subroutines, update the provided *makefile.** build script with the object file names of any new subroutines. When the *makefile.** build script runs, it creates an executable file called *dr3serv*. Make sure all files referenced in *makefile.** are in the same directory as *dr3serv*, or point to the necessary directory in *makefile.**.

3. Add a CONNECT statement to the input file

   Add a CONNECT File Management statement to the NX Nastran input file. The CONNECT statement creates the relationship between the DRESP3 Bulk Data entries and the executable file. The format of the CONNECT statement is:

   CONNECT   DRESP3   GROUP   EVALUATOR

   where:

   - The GROUP field on the DRESP3 Bulk Data entries should match the GROUP field on the CONNECT statement.

- The EVALUATOR field on the CONNECT statement specifies the alias for the executable file.

In the following example, the group name of “TESTGRP” on the CONNECT statement is used in the GROUP field of both DRESP3 Bulk Data entries. Therefore both DRESP3 entries will use the same executable file. However, the DRESP3 Bulk Data entry with 31 as the ID uses “EULER” as the response type, and the DRESP3 Bulk Data entry with 32 as the ID uses “JOHNSON” as the response type. These different response types refer to two different response calculations in the executable file. The results of both DRESP3 responses in this example are then used in design constraints.

```
CONNECT DRESP3 TESTGRP EXTRESP
...
SOL 200
...
DESVAR  1       RG      1.0     0.01    10.0
DRESP1  25      S1      STRESS  PBAR            6               10
DRESP3  31      EULER   TESTGRP EULER
        DESVAR  1
        DTABLE  L       E
        DRESP1  25
DRESP3  32      JOHNSON TESTGRP JOHNSON
        DESVAR  1
        DTABLE  L       E       SIGMAC
        DRESP1  25
DCONSTR 1       31      1.0
DCONSTR 1       32      1.0
...
```

4. Define the alias for the executable file

   Create an evaluator path file that relates the alias name in the EVALUATOR field of the CONNECT statement to the actual path of the executable file. For the above example, “EXTRESP” is the alias and *dr3serv* is the executable file. Thus, the evaluator path file would include the following line of text:

   EXTRESP,-,/network_drive/user_dir/dr3srv/dr3serv

5. Use the GMCONN keyword to reference the evaluator path file

   You assign the GMCONN keyword to the name of the evaluator path file. For example, suppose the evaluator path file you created in the previous step is named *nasevals*. The command line syntax would be:

   .../nxnr/bin/nastran.exe gmconn=/network_drive/user_dir/dr3srv/nasevals input_file.dat

Format of Data Passed to the Executable File

The data included on a DRESP3 Bulk Data entry is passed to the executable file in three one-dimensional arrays:

- The “arglis” array contains the number of each type of entry on the DRESP3 Bulk Data entry.
- The “argval” array contains the numerical value or character string that corresponds with each entry listed on the DRESP3 Bulk Data entry.
- The “argchar” array contains character strings for the USRDATA entries included on the DRESP3 Bulk Data entry.

An example is useful for demonstrating how the “arglis” and “argval” arrays are formatted. Suppose a DRESP3 entry contains two DESVAR entries, one DRESP1 entry, and six DVPREL1 entries. The “arglis” array would be:

2
0
0
1
6

The zeros in the “arglis” array designate that there are no DTABLE or DFRFNC entries. Likewise, there are no DVPREL1, DVCREL1, DVMREL1, DVPREL2, DVCREL2, DVMREL2, and DRESP2 entries. However, unlike DTABLE and DFRFNC, zeros are not required for the DVPREL1, DVCREL1, DVMREL1, DVPREL2, DVCREL2, DVMREL2, and DRESP2 entries. The executable the build script creates will automatically interpret any omitted rows at the bottom of the “arglis” array as zero.

The corresponding “argval” array contains the numerical values and character strings associated with the nonzero entries in the “arglis” array. For this example, the “argval” array contains nine entries – two attributable to DESVAR entries, one attributable to a DRESP1 entry, and six attributable to DVPREL1 entries. The nine entries in the “argval” array would be ordered as follows:

DESVAR Value 1 (DVID1 on the DRESP3 entry)
DESVAR Value 2 (DVID2 on the DRESP3 entry)
DRESP1 Value (NR1 on the DRESP3 entry)
DVPREL1 Value 1 (DPIP1 on the DRESP3 entry)
DVPREL1 Value 2 (DPIP2 on the DRESP3 entry)
DVPREL1 Value 3 (DPIP3 on the DRESP3 entry)
DVPREL1 Value 4 (DPIP4 on the DRESP3 entry)
DVPREL1 Value 5 (DPIP5 on the DRESP3 entry)
DVPREL1 Value 6 (DPIP6 on the DRESP3 entry)

The actual “argval” array passed to the executable will contain a numerical value or character string for each entry.

In this example, there are no USRDATA entries. However, the “argchar” array and its length NWRDA8 should still be entered into the argument list of either *dr3svald.F* (for LP) or *dr3svals.F* (for ILP). This is the case because NX Nastran calls one of these subroutines and *r3sgrt.F* to determine whether the specified response is a valid one.

Design Responses and Case Control

You can reference design responses from the Case Control section in two ways:

- By using subcase-based constraint and/or objective function definitions that you specify with DESSUB or DESOBJ Case Control commands, respectively, within a subcase.
  - o DESSUB refers to the constraint definition DCONSTR bulk data entry (or DCONADD bulk data entry which specifies multiple DCONSTR entries), which, in turn, references the appropriate DRESPi (DRESP1, DRESP2, or DRESP3) entry or entries.
  - o DESOBJ directly references the relevant DRESPi bulk data entry or entries.
- By using global and/or objective function definitions that you specify with DESGLB and DESOBJ Case Control commands that appear in the Case Control section above any subcases. However, if such DESGLB or DESOBJ commands indirectly point to a DRESP1 entry assigned to specific subcases with a DRSPAN command, the software uses them to combine responses from various subcases.

  For example, consider an analysis in which a DESGLB commands points to a DCONSTR constraint entry on a DRESP2 entry that combines two displacement DRESP1 responses. For this to be valid, the two DRESP1 entries must be assigned to subcases using sets referenced by DRSPAN commands in the relevant subcases for which they are to be computed. For more information, see DRSPAN, DRESP2, and DRESP3 in the *NX Nastran Quick Reference Guide*.

Case Control and Design Response Form

Structural responses may often be computed in different forms. Dynamic responses may either be expressed using complex quantities or in a magnitude phase representation. Plate element strains may be output in strain-curvature or strain-at-fiber form. Where these types of options exist, data recovery in design optimization will use the Case Control Section to determine the output form. If no Case Control requests are present, the default representations are used. (These are listed in the *NX Nastran Quick Reference Guide*.)

For example, von Mises stresses are the default form for NX Nastran output. This is taken as the default for design optimization as well. Thus, in the absence of any Case Control requests to the contrary, the following will select von Mises stresses for all elements in property group 150:

```
$DRESP1,ID,     LABEL,   RTYPE,   PTYPE,   REGION, ATTA,    ATTB,    ATT1,    +
$+,     ATT2,    ...
DRESP1, 100,     SIG1,    STRESS, PSHELL, ,       9,       ,        150
```

The integer 9 in the ATTA field selects the von Mises stress component as surface Z1 for the shell elements.

However, the following may be used to select stresses in maximum shear rather than von Mises form:

```
$  (in case control...)
STRESS (SHEAR) = ALL
$  (in bulk data...)
$DRESP1,ID,     LABEL,   RTYPE,   PTYPE,   REGION, ATTA,    ATTB,    ATT1,    +
$+,     ATT2,    ...
DRESP1, 100,     SIG1,    STRESS, PSHELL, ,       9,       ,        150
```

Note that the DRESP1 entry is unchanged, with the response form determined solely by Case Control.

Case Control may also be used to limit output in frequency response and transient response analysis, using the OFREQ and OTIME commands. This can be used to reduce the cost of design sensitivity analysis as is discussed in the next section.

Identifying Dynamic Responses

In Frequency Response analysis, a number of forcing frequencies may be used. Similarly, a number of analysis time steps may be used in a transient analysis. Both of these analysis types have the potential to generate a huge amount of design response data. This could get quite expensive, as sensitivities (discussed in Example Problems) may need to be computed for all of these design responses.

Limiting the Number of Design Responses with OTIME and OFREQ

To keep this data to a minimum, the OFREQ and OTIME Case Control commands can be used to limit the number of design responses. For frequency response, design responses are only computed for those frequencies referenced by the OFREQ command. Otherwise, design responses will be computed for all forcing frequencies by default.

Note

Time steps can be stated explicitly in the ATTB field of the DRESP1 entry. If this value is not equal to any of the time values shown in Figure 2-25, then the nearest available reduced time step in the figure will be used.

For transient response, design responses are only computed for the output time steps. Output time steps are defined by the NOi fields on the TSTEP Bulk Data entry and may be further limited by the OTIME Case Control command. In the absence of any NOi or OTIME definitions, design responses are computed for all analysis time steps. Figure 2-25 is a schematic that indicates the levels of time step reduction.

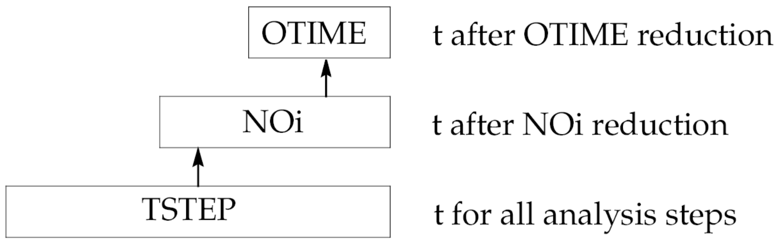


Figure 2-25. Output Time Step Reduction.

Similarly, for frequency response, if the value of a frequency in the ATTB field is not equal to any of the frequencies in the relevant frequency set for the applicable subcase, NX Nastran replaces the value in the ATTB field with the nearest frequency in the set.

Design responses can also be computed for a single forcing frequency, or for a single time step. We may want to define a design response as a displacement average over time. Having identified a peak transient response, this technique could help us to reduce this overshot. (Often, as a design is modified, the transient peak will “move around” some from one design cycle to the next. Computing an average, rms, or other measure over time or frequency, will often yield more useful results.)

The following example illustrates the formation of an average displacement over five time steps. This average is based on the z-component of displacement for grid 100.

```
$ Identify component displacements:
$DRESP1,ID,     LABEL,  RTYPE,  PTYPE,  REGION, ATTA,   ATTB,   ATT1,   +
$+,     ATT2,   ...
DRESP1, 11,     Z051    TDISP,  ,       ,       3,      .051,   100
DRESP1, 12,     Z052    TDISP,  ,       ,       3,      .052,   100
DRESP1, 13,     Z053    TDISP,  ,       ,       3,      .053,   100
DRESP1, 14,     Z054    TDISP,  ,       ,       3,      .054,   100
DRESP1, 15,     Z055    TDISP,  ,       ,       3,      .055,   100
$
$ Generate average displacement over time...
$DRESP2,ID,     LABEL,  EQID,   REGION, ,       ,       ,       ,       +
$+,     DESVAR, DVID1,  DVID2,  ...,    ,       ,       ,       ,       +
$+,     DTABLE, LABEL1, LABEL2, ...,    ,       ,       ,       ,       +
$+,     DRESP1, NR1,    NR2,    ...,    ,       ,       ,       ,       +
$+,     DNODE,  NID1,   DIR1,   NID2,   DIR2,   ...,    ,       ,       +
$
DRESP2, 21,     AVG,    200,    ,       ,       ,       ,       ,       +
+,      DRESP1, 11,     12,     13,     14,     15
$
$ ...using the equation input capability
$DEQATN EQUID   F() = ...
DEQATN  200     AVERAGE(U1,U2,U3,U4,U5) = (U1+U2+U3+U4+U5)/5.
```

Note

If response values change sign, an RMS average may yield more useful results.

DRESP1 entries 11 through 15 identify the z-component of displacement at grid 100 for time steps .051, .052, .053, .054, and .055. Note that each entry defines a single displacement component at a single grid for a single time step. The DRESP2 defines the second-level response by identifying each of these individual responses as input to a DEQATN entry. The resulting second-level response can either be used as an objective or constraint.

2.7 Defining the Objective Function

The objective function is a scalar quantity that is either minimized or maximized by the optimizer. You define the design objective using the DESOBJ (DESign OBJective) Case Control command. This command points to a design response defined on a DRESP1, DRESP2, or DRESP3 Bulk Data entry that must define a single scalar response only.

Formulating the Objective: Sensitivity with Respect to Design Variables

When formulating the design objective, there are a couple of scaling-related issues that should be kept in mind since they affect overall performance. First, the design problem should be posed so that the objective function has sufficient sensitivity with respect to each of the design variables. This is a relative type of consideration and is somewhat difficult criteria to define in a general sense. However, the idea is as follows: Suppose a weight design objective is on the order of 1,000 kilograms. If a structural element in the model is on the order of 1.0E-3 kg or less and its volume is a function of a single design variable, then varying this quantity by 100% or more does not have any appreciable effect on the overall weight. It is probably a better idea either to link a number of these types of elements together and relate them all to a single independent variable (resulting in a larger weight change) or to eliminate these element groups entirely from the redesign process. Prior to optimization, it is always advisable to perform a design sensitivity analysis first to ensure that the objective function has sufficient sensitivity with respect to the design variables.

Formulating the Objective: Magnitude

The second item to consider is the absolute value of the response selected to be the objective function. Care should be taken that this value is not too close to zero. Here again it is difficult to define what range of values is too small, but the absolute value of the objective should be greater than about 1.0 if possible. If this requirement is not met, there are a couple of ways in which this may pose problems.

Note

Does your design “converge” after only a single design cycle? If so, and your initial objective function value is on the order of 1.0E-2 or smaller, CONV2 (on the DOPTPRM entry) should be reduced.

For very small absolute values of the objective function, the default convergence criteria will not be satisfactory. The convergence criteria logic is detailed in Convergence Tests, but of interest here is the test on absolute change in the objective function. This default is 0.01, but can and should be changed using a DOPTPRM (Design OPTimization PaRaMeters) Bulk Data entry. With the default, if the objective function changes by less than 0.01 (absolute) from the previous design cycle and all constraints are satisfied, the process has converged. This may not be a reasonable test if the original objective function is small.

The optimizer itself may also run into numerical difficulties for small absolute values of the objective function. This situation may affect the optimizer’s estimates of an initial value of α for the one-dimensional search as well as the convergence criteria at the optimizer level. Again, the defaults can be changed using the DOPTPRM entry, although determining correct values for the override may not be easy. It may be far better to rescale the problem, multiply the objective by 100, or other such avoidance, rather than try to change the optimizer’s defaults.

Formulating the Objective: Sign Changes

If the objective function is a response, it is important that you know whether the response is expected to change signs for different designs that may be generated during the optimization. For example, minimizing a positive displacement means driving it in the direction towards zero. However, if the displacement changed sign during the optimization, then the same minimization would try to drive it towards negative infinity. In such cases, it may be better to use a device that would minimize the absolute value through a DRESP2 entry as applicable, such as the square root of the square or of the sum of the squares.

2.8 Defining the Constraints

Any first-or second-level response may be constrained simply by referencing it on a DCONSTR (Design CONSTRaint) Bulk Data entry. This entry defines the response to be constrained and its corresponding bounds.

To define constraints in NX Nastran, you need to

- Identify a first-level response (or group of responses) using a DRESP1 Bulk Data entry or write a second-level response (or group of second-level responses) using a DRESP2 or a DRESP3 Bulk Data entry. If the DRESP2 or DRESP3 entry references a DRESP1 entry, the DRESP1 entry can be a generic DRESP1 entry or a DREPS1 entry assigned to specific subcases with a DRSPAN Case Control command. If the DRESP2 or DRESP3 entry reference a DRESP1 entry that is assigned with a DRSPAN Case Control command, then all DRESP1 entries referenced by the DREPS2 or DRESP3 entries must be assigned with a DRSPAN command.

- Specify the response bounds by using a DCONSTR entry that references the DRESP1, DRESP2, or DREPS3 defined response or set of responses, for example, if it is defined over a frequency set.

- Select DCONSTR entry sets in the Case Control section using either the DESGLB command for "global" (DRSPAN related, therefore subcase dependent, or not DRSPAN related, therefore subcase independent) responses or the DESSUB command for subcase-dependent responses. Responses or constraints become subcase-dependent when any related DRESP1 entries are referenced from a DCONSTR entry that is, in turn, referenced from a SESUB command belonging to a subcase, rather than assigned with a DRSPAN command. Therefore, you can reference the non-DRSPAN related responses from more than one subcase by using differently numbered DCONSTR entries that are referenced from DESUB commands from different subcases.

The DCONSTR entry bounds are used by NX Nastran to create a pair of normalized constraints—one for the lower bound and one for the upper bound. For any given response, the bounds are a statement of the inequality relation

$$r_j^L \le r_j(\vec{x}) \le r_j^U$$

Equation 2-34.

where $r_j{}^L$ is the lower bound on the j -th response and $r_j{}^U$ is the corresponding upper bound.

Normalized Constraints

The absolute values of the response bounds are used as normalizing factors in the resulting inequality constraints

$$g_{2j-1}(\vec{x}) = \frac{r_j^L - r_j(\vec{x})}{\left|r_j^L\right|} \le 0$$
$$g_{2j}(\vec{x}) = \frac{r_j(\vec{x}) - r_j^U}{\left|r_j^U\right|} \le 0$$

Equation 2-35.

that are of the same inequality form as the constraints in the basic optimization problem statement.

Physical Significance of Normalized Constraints

Normalized constraints are especially useful since the dependence on the magnitude of the response quantity has been removed. A constraint having a value of +1 represents a 100% violation irrespective of the magnitude of the response or whether it is an upper or lower bound. Normalization is used in the constraint deletion phase to temporarily screen out constraints that are not active.

An overview of constraint deletion is given in Approximation Concepts in Design Optimization in connection with approximation techniques.

Using GSCAL to Provide Minimum Constraint Normalizing Factors

Since response bounds are used as normalizing factors, it is important to avoid specifying the bounds as zero. To avoid a divide by zero error in the code, a small nonzero number is used instead, which may or may not be acceptable from a design standpoint. is provided by the parameter GSCAL, which can be defined on the DOPTPRM entry. Its default is 0.001. If either of the normalizing factors $|r_j^L|$ or $|r_j^U|$ are less than GSCAL, GSCAL is used instead. For example, consider the synthetic response

$$\frac{\sigma_1 + \sigma_2}{2} \leq \sigma_{max}$$

Equation 2-36.

and assume the response had been formulated as

$$\frac{\sigma_1 + \sigma_2}{2} - \sigma_{max} \leq 0$$

Equation 2-37.

At first glance, this seems acceptable since it is expressed in standard form (see The Basic Optimization Problem Statement). However, the zero upper bound is replaced by a nonzero ε and the resultant normalized upper-bound constraint then becomes

$$g_u = \frac{\left(\frac{\sigma_1 + \sigma_2}{2} - \sigma_{max}\right)}{\varepsilon} \leq 0$$

Equation 2-38.

The derivative of this constraint with respect to changes in one of the stress responses is

$$\frac{\partial g_u}{\partial \sigma_1} = \frac{1}{2\varepsilon}$$

Equation 2-39.

which is a large quantity due to the small ε. Numerically, the optimizer considers a constraint to be active only if it is greater than CT (constraint tolerance) and less than CTMIN (minimum allowable constraint violation). If a constraint is greater than CTMIN, it is considered violated. This suggests that the constraint as expressed by Eq. 2-39 is poorly conditioned since, as a result of the derivative's magnitude, the region over which this constraint is active is relatively small. The situation is shown in Figure 2-26.

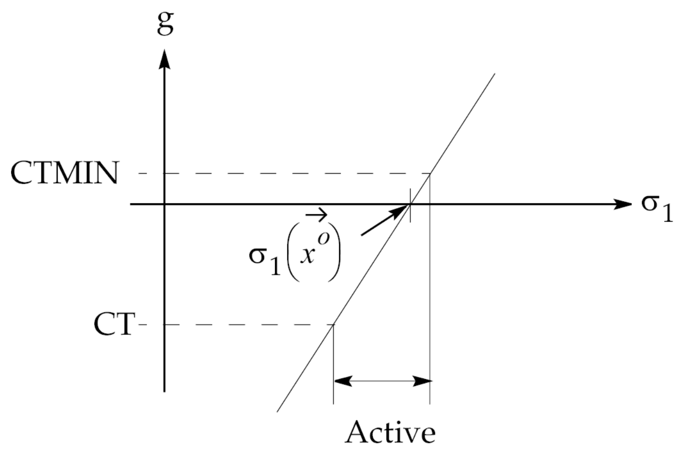


Figure 2-26. Poorly-Conditioned Constraint

If the constraint is expressed instead as

$$\frac{\sigma_1 + \sigma_2}{2} \le \sigma_{max}$$

Equation 2-40.

then the limit σ_{max} (provided as an upper bound on the DCONSTR entry) is used as the normalizing factor or

$$g_u = \frac{\left(\frac{\sigma_1 + \sigma_2}{2}\right) - \sigma_{max}}{\sigma_{max}} \le 0$$

Equation 2-41.

The derivative of this constraint with respect to changes in σ^1 is now

$$\frac{\partial g_u}{\partial \sigma_1} = \frac{1}{2\sigma_{max}}$$

Equation 2-42.

As is shown in Figure 2-27, the region over which this constraint is active is now much greater than the previous formulation. This change improves the conditioning of the problem since it is less likely that active constraints quickly become either inactive or violated as the search progresses.

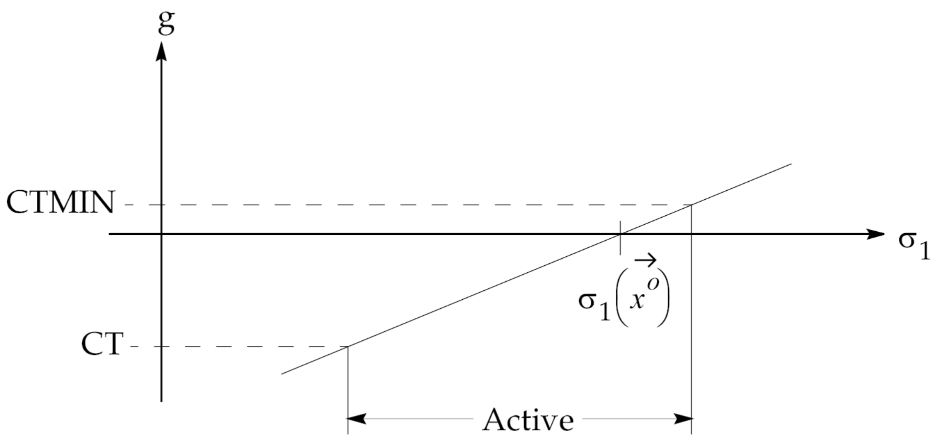


Figure 2-27. Well-Conditioned Constraint

Of course, a second-level response can be written so that it is already in normalized form. The synthetic response in Eq. 2-36 could have been written as

$$\frac{\sigma_1 + \sigma_2}{2\sigma_{max}} \leq 1$$

Equation 2-43.

This would give the same results as shown in Figure 2-27. Either normalized formulation is acceptable.

Equality Constraints in NX Nastran

You can define an equality constraint simply by specifying equivalent upper and lower bounds on a DCONSTR entry. This will yield a pair of equal and opposite constraints that force the response to be satisfied with equality at the optimum:

$$R_j \leq r_j(\vec{x}) \leq R_j$$

$$g_{2j-1}(\vec{x}) = \frac{R_j - r_j(\vec{x})}{|R_j|} \leq 0$$

$$g_{2j}(\vec{x}) = \frac{r_j(\vec{x}) - R_j}{|R_j|} \leq 0$$

Equation 2-44.

Often in design we are not concerned with satisfying an equality constraint precisely; rather any satisfaction within a certain tolerance will do. (In fact, attempting to satisfy a number of equality constraints precisely may be a problem for the optimizer.) We may want to match frequencies, for example, to within ±1% of some target values. This could be expressed using lower and upper bounds of $R_j - \delta$ and $R_j + \delta$ on the DCONSTR entry, where δ is the allowable tolerance.

2.9 Superelement Design Modeling

With few exceptions, design modeling for superelement sensitivity and optimization is a largely transparent operation in Solution 200. That is, design models spanning multiple superelements can be expressed using the same tools and methods for non-superelement models discussed in the previous sections. This is a useful tool in any large-scale design task.

This section introduces the differences in superelement versus non-superelement design modeling. Limitations are summarized at the end of the section.

Supported Superelements

Primary and image superelements can only be used in connection with the design model. Primary, image, and external superelements can all be used in the analysis model.

The external superelement restriction occurs because external superelements are only known to the code via their structural stiffness, mass, damping, and load matrices. Not having any of the properties or other modeling data on hand prevents referencing any of this data in connection with the design model. Although external superelements may be used in connection with the analysis model, their properties are assumed to be constant with respect to the design model.

Design Variables in Superelement Design Modeling

You can relate design variables to properties as in the case with non-superelement models by using either linear relationships specified on DVPREL1 Bulk Data entries or user-defined equations on DVPREL2 entries. Design variable-to-property relations can be local to a given superelement or can span superelement boundaries. For example, a design variable can be related to a particular property entry which is, in turn, used in a number of superelements in the model.

Shape Basis Vector Restrictions

You can also define design variable-to-grid relationships using DVGRID Bulk Data entries. A given design variable can be related to grid sets that span superelement boundaries. A DVGRID entry may reference both interior and exterior grids of primary superelements. DVGRID entries are the only way to specify shape basis vectors in superelement design models. The other methods—direct input of shapes, geometric boundary shapes, and analytic boundary shapes—are not supported.

For secondary superelements, the same property and/or grid variations prescribed on the primary superelement also apply to the corresponding image superelements. All references to external grids of secondary superelements are ignored, however. This stems from the fact that the superelement mapping matrix is used to compute the secondary superelement matrices using the primary superelement.

Design Responses in Superelement Design Modeling

You can select design responses using DRESP1, DRESP2, or DREPS3 Bulk Data entries or using a combination of those entries. A single DRESP1 entry can be used to select a number of design responses by referring to multiple grid locations, property IDs, or elements. The responses identified on a single DRESP1 entry can be from multiple superelements.

For weight and volume type responses, the applicable superelement IDs (SEIDi), or ‘ALL’ must be specified. Weight or volume responses can be computed for the entire model or just a subset defined by superelement ID reference. The default is “ALL”, which is the total weight/volume of all superelements except external superelements.

Note

Weight and volume sensitivities can be computed on an individual superelement basis, or for ‘ALL’ superelements.

DRESP2 and DRESP3 Restrictions

The design responses input to DRESP2 equations or to DRESP3 external programs cannot span multiple superelements or load cases (unless they are DRSPAN related). All synthetic responses are checked to ensure that all response arguments are defined via DRESP1 entries and belong to the same superelement. If any of these criteria are violated, a fatal error message is issued.

Superelements and Constraint Screening

For superelement sensitivity, the constraint screening criteria are applied on an individual superelement basis. That is, if a DRESP1 entry lists several element, grid, or property IDs that span superelement boundaries, the screening criteria will apply individually to all superelements that share responses listed on this DRESP1 entry. In other words, regions are subsetted by superelement. If separate sets of constraint screening parameters, TRS (truncation threshold), and NSTR (number of retained constraints per region) are required for different superelements, then separate sets of DRESP1 and DCONSTR entries should be used to define superelement-specific response groups.

Case Control

The simplest way to incorporate superelements is to use the SUPER = ALL command in Case Control. This automatically satisfies a number of design optimization Case Control requirements. You can also use an expanded subcase structure if necessary, with each subcase pertaining to a superelement or group of superelements. If you choose the latter approach there are some additional requirements that must be satisfied:

- Each superelement subcase must define an analysis type using the ANALYSIS=command.
- For ANALYSIS = MODES, the eigenvalue constraint must be called out from each MODES subcase using the DESSUB command.
- The METHOD command included in the residual subcase must also be used in the upstream superelement subcases. This requirement also holds for Component Modes Synthesis (CMS).

Limitations

The limitations that apply to superelement design modeling are summarized as follows:

- The design model may not reference external superelements. External superelements can be part of the analysis model but are considered invariant with respect to changes in the design model.
- Image superelements have the same design variations and design responses as defined on the referenced primary superelement. The exception is external grids, which are invariant for the image superelements.
- Shape basis vectors can only be defined using the DVGRID entry.

- Eigenvalue sensitivity is evaluated for the residual superelement. Since the residual contains the boundary degrees of freedom from all superelements, the computed eigenvalue sensitivities pertain to the entire structure. Eigenvalues computed during upstream superelement reduction are not available as design responses.

  Note

  In addition, Guyan reduction (the default) is exact for stiffness properties but only approximate for mass. Thus, you may want to consider using advanced reduction methods such as component modes synthesis to improve the sensitivity analysis accuracy.

- Synthetic responses defined via DRESP2 and DRESP3 entries cannot have arguments that span superelement boundaries.

- Multiple boundary conditions are not permitted in superelement design modeling. This restriction stems from a similar restriction in superelement analysis.

- The DCONSTR and DSCREEN entries have no provisions for specification of superelement IDs. If separate response allowables or screening criteria are to be applied, the DRESP1 entries should define responses that do not span superelement boundaries.

Chapter 3: Design Sensitivity and Optimization in NX Nastran

This chapter presents the theoretical details of design sensitivity and optimization in NX Nastran. Intended as a complement to the design modeling topics of the previous chapter, each section begins with a general overview of modeling-related aspects before moving on to the theoretical details. The presentation order of this chapter generally follows the code's order of operations.

3.1 Approximation Concepts in Design Optimization

Approximation concepts are used in NX Nastran to efficiently couple the numerical optimizer with the structural analysis code. This section discusses the need for and the theory behind the approximations available in NX Nastran. You can control the form of these approximations by

- Specifying the approximation method using the DOPTPRM entry (APRCOD = 1, 2, or 3).
- Providing explicit design variable linking with DLINK Bulk Data entries.
- Controlling the regionalization of responses using DRESP1, DRESP2, and DRESP3 entries, and the constraint screening criteria using DSCREEN entries.

Why Are Approximations Necessary in Design Optimization?

As outlined in Structural Optimization, approximation concepts address three fundamental difficulties associated with structural optimization:

- There are often more design variables than are necessary to adequately describe the allowable design variations.
- A typical structural optimization problem often contains hundreds or even thousands of constraints. It is likely that many of these constraints will contain redundant or irrelevant design information.
- The structural responses are only implicit functions of the design variables. Having to repeatedly invoke a full finite element analysis to evaluate these implicit functions for small design changes is not cost-effective.

Approximations in NX Nastran: An Overview

The preceding difficulties are effectively resolved with approximation concepts. In NX Nastran, there are essentially three categories into which these approximations fall. These categories and the control you have over each are first itemized below. The remainder of the section discusses each of these items in greater detail.

- Design variable linking:

  Design variable linking allows you to keep the number of independent design variables to a minimum, leading to a well-formulated design model. You link design variables as part of the design modeling process. Any combination of three possible methods can be used: grouping of the analysis model properties, effective use of the DVPREL1 and 2 relations, and explicit linking of the design variables via the DLINK entry. In addition to reducing the computational overhead (both during sensitivity analysis as well as optimization), design optimization results interpretation is usually simplified.

Note

Design variable linking not only reduces the cost of design sensitivity and optimization, it also makes results interpretation easier.

- Constraint regionalization and deletion:

  This process temporarily removes non-critical, redundant constraints from consideration on the assumption that this reduced set still contains adequate information to efficiently guide the design. NX Nastran performs this task automatically, although you do have some high-level control of the process. You can use the DSCREEN, DRESP1, and DRESP2, and DRESP3 entries to provide this constraint screening control.

  Note

  Constraint deletion reduces the cost of sensitivity analysis by temporarily deleting design constraints and the associated responses. The sensitivity of these responses does not need to be computed for the current design cycle. Formal approximations allow the optimizer to make design changes without having to invoke a full finite element analysis each time.

- Formal approximations:

  Formal approximations allow the optimizer to make design changes without having to invoke a full finite element analysis each time.

  In the code, the objective function and all retained constraints are cast in terms of high-quality approximations that are explicit in the design variables. The optimizer refers to these approximations when it requires function evaluations instead of the costly and implicit finite element analysis. These approximations are constructed automatically in NX Nastran according to one of three possible methods. (This choice is problem-dependent.) You can choose from direct linearization, mixed method, and convex linearization. The method is selected by setting APRCOD to 1, 2, or 3 using the DOPTPRM Bulk Data entry. The mixed method (APRCOD = 2) is the default.

Approximate Model

Figure 3-1 illustrates the connection between the finite element analysis, the approximate model, and the optimizer. Note that the approximate model, constructed using the results of the finite element analysis, provides the optimizer with the design information it requires instead of directly from the finite element analysis. The approximate model in Figure 3-1 incorporates the effects of the reduced set of design variables, the screened set of constraints, and the approximated set of structural responses.

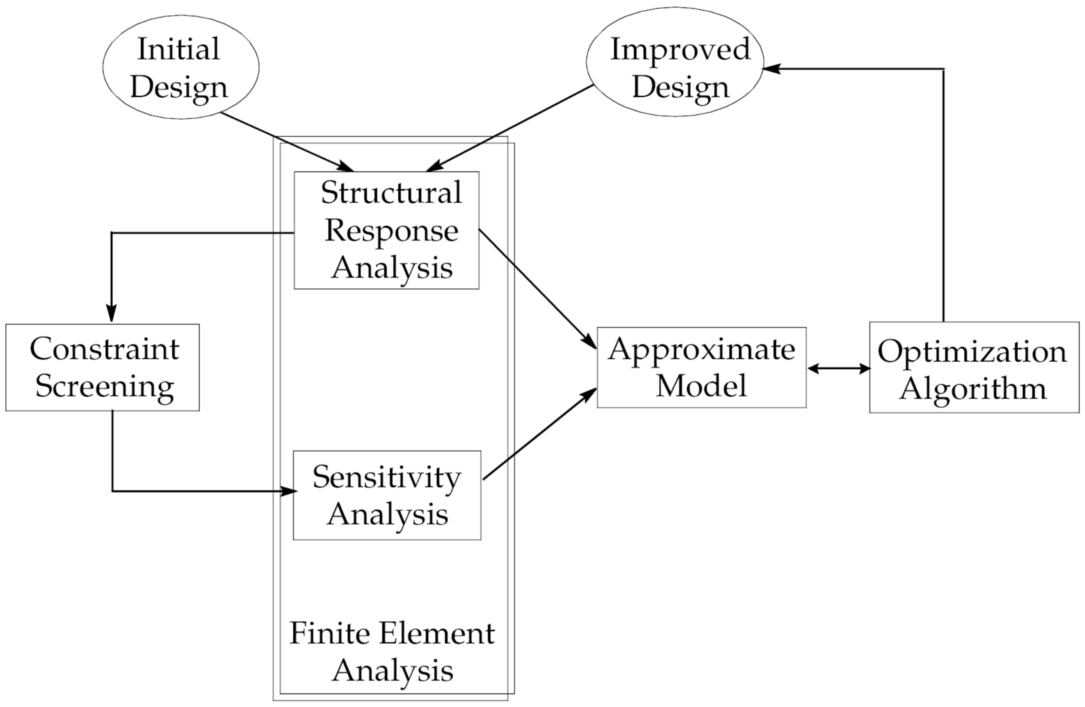


Figure 3-1. Approximation Concepts

Design Variable Linking

The optimizer improves the design by changing the values of the design variables. The complexity (and computational cost) of the overall process tends to increase as the number of design variables is increased. One of your design modeling goals should be to describe the permissible design changes using as few design variables as necessary. (A problem consisting of 200 to 300 independent design variables is currently considered a large design task in NX Nastran.) On the next few pages is a discussion of the methods that you can use to achieve this goal.

Linking by Analysis Model Properties

One of the more efficient methods you can use to minimize the number of independent design variables is to consolidate the number of analysis model properties that are referenced in the design model.

The situation is shown in Relating Design Variables to Properties, Figure 2-3. A DVPRELi entry (i = 1 or 2) defines a design variable-to-property relation for a particular property entry identified by its property ID, or PID. A large number of elements, in turn, might reference this property entry. Since they all reference the same PID, all elements in this group will undergo equal variations as the design variables are changed.

By considering the design implications when defining property groups, you may be able to reduce their numbers. A smaller number of designed property groups might then require fewer independent design variables, since groups of elements have already been "linked" by their property groups memberships.

Linking with Linear Relations

If the number of analysis model property entries cannot be reduced, it may be worthwhile to consider explicit methods of reducing the number of independent design variables. One way you can do this is by using linear design variable-to-property relations.

The DVPREL1 Bulk Data entry is used to expresses an analysis model property as a linear function of design variables or

$$p_i = c_0 + c_1 x_1 + c_2 x_2 + \ldots + c_n x_n$$

Equation 3-1.

In this equation, the *i*-th property p_i is a function of the set of design variables x_1 through x_n. One feature of Eq. 3-1 is that it can be used to express a large number of properties in terms of a much smaller set of design variables. This is shown in the following example.

Example

Assume an optimal plate thickness distribution is to be determined using a combination of constant, linear, and quadratic basis functions as shown in the figure. The design variables act as multipliers of these basis functions that are, in turn, used to specify the element plate thicknesses.

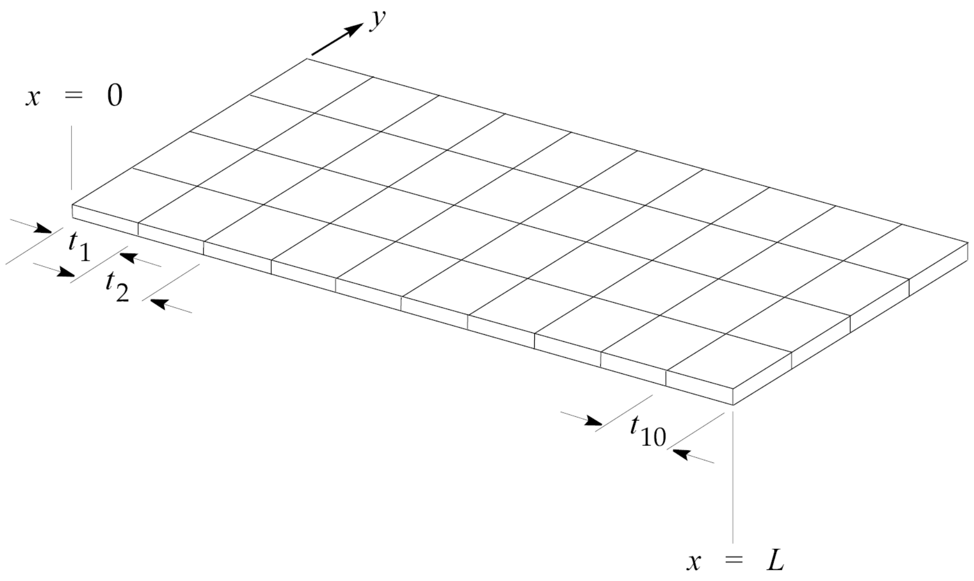
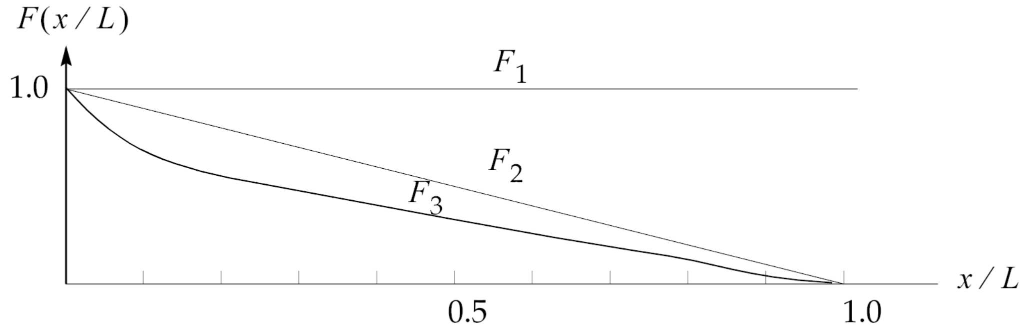


Figure 3-2. Regions of Validity for the Plate Thickness



$$F_1\left(\frac{x}{L}\right) = C_1$$

$$F_2\left(\frac{x}{L}\right) = C_1\left(1 - \frac{x}{L}\right)$$

$$F_3\left(\frac{x}{L}\right) = C_1\left(1 - \frac{x}{L}\right)^2$$

$$C_1 = 1.0$$

Figure 3-3. Basis Functions

Suppose the plate thicknesses are to vary in the x-axis direction, but are to remain constant in the y-direction. Evaluating the basis functions for each of these ten thickness stations and writing these design variable-to-thickness relations in matrix form, we get

$$\begin{Bmatrix} t_1 \\ t_2 \\ t_3 \\ t_4 \\ t_5 \\ t_6 \\ t_7 \\ t_8 \\ t_9 \\ t_{10} \end{Bmatrix} = \begin{bmatrix} 1.0 & 1.0 & 1.0 \\ 1.0 & 0.9 & 0.81 \\ 1.0 & 0.8 & 064 \\ 1.0 & 0.7 & 0.49 \\ 1.0 & 0.6 & 0.36 \\ 1.0 & 0.5 & 0.25 \\ 1.0 & 0.4 & 0.16 \\ 1.0 & 0.3 & 0.09 \\ 1.0 & 0.2 & 0.04 \\ 1.0 & 0.1 & 0.01 \end{bmatrix} \begin{Bmatrix} X_1 \\ X_2 \\ X_3 \end{Bmatrix}$$

Equation 3-2.

Reduced Basis Formulations

The thicknesses of the ten element groups are controlled using only three design variables, X_1, X_2, and X_3. Each of the matrix columns correspond to constant, linear, and quadratic basis functions, respectively. Each column is a vector, and the dimension of each of these vectors is greater than the number of such vectors. Relations of this sort are termed reduced basis formulations.

In general, a reduced basis formulation can be expressed as

$$\{p\}_M = [T]_{MXN}\{x\}_N$$

Equation 3-3.

The most powerful form of Eq. 3-3 occurs when the number of rows of [*T*] is much greater than the number of columns or *M* >> *N*. This concept is used extensively in connection with shape optimization; a large set of grid coordinate variations can be governed by a much smaller set of design variables (see Relating Design Variables to Shape Changes).

Listing 3-1 shows the Bulk Data entries that can be used to implement the design formulation of Eq. 3-2. Since the reduced basis formulation is just a set of linear equations, DVPREL1 Bulk Data entries can be used exclusively. Also, since each of the plate strips are of constant thickness, each element group shares a property Bulk Data entry for a total of ten such entries over 40 plate elements.

```
$PSHELL,PID,    MID1,   T,      MID2,   12I/T3  MID3
PSHELL, 101,    100,    1.0,    100,    ,       100
PSHELL, 102,    100,    1.0,    100,    ,       100
PSHELL, 103,    100,    1.0,    100,    ,       100
PSHELL, 104,    100,    1.0,    100,    ,       100
PSHELL, 105,    100,    1.0,    100,    ,       100
PSHELL, 106,    100,    1.0,    100,    ,       100
PSHELL, 107,    100,    1.0,    100,    ,       100
PSHELL, 108,    100,    1.0,    100,    ,       100
PSHELL, 109,    100,    1.0,    100,    ,       100
PSHELL, 110,    100,    1.0,    100,    ,       100
$
$DESVAR,ID,     LABEL,  XINIT,  XLB,    XUB,    DELXV
DESVAR, 1,      X1,     0.33,   -1.0,   +1.0
DESVAR, 2,      X2,     0.33,   -1.0,   +1.0
DESVAR, 3,      X3,     0.33,   -1.0,   +1.0
$
$DVPREL1,ID,    TYPE,   PID,    FID,    PMIN,   PMAX,   C0,     ,       +
$+,     DVID1,  COEF1,  DVID2,  COEF2,  ...
DVPREL1,201,    PSHELL, 101,    4,      0.01,   ,       ,       ,       +
+,      1,      1.0,    2,      1.0,    3,      1.0
DVPREL1,202,    PSHELL, 102,    4,      0.01,   ,       ,       ,       +
+,      1,      1.0,    2,      0.9,    3,      0.81
DVPREL1,203,    PSHELL, 103,    4,      0.01,   ,       ,       ,       +
+,      1,      1.0,    2,      0.8,    3,      0.64
DVPREL1,204,    PSHELL, 104,    4,      0.01,   ,       ,       ,       +
+,      1,      1.0,    2,      0.7,    3,      0.49
DVPREL1,205,    PSHELL, 105,    4,      0.01,   ,       ,       ,       +
+,      1,      1.0,    2,      0.6,    3,      0.36
DVPREL1,206,    PSHELL, 106,    4,      0.01,   ,       ,       ,       +
+,      1,      1.0,    2,      0.5,    3,      0.25
DVPREL1,207,    PSHELL, 107,    4,      0.01,   ,       ,       ,       +
+,      1,      1.0,    2,      0.4,    3,      0.16
DVPREL1,208,    PSHELL, 108,    4,      0.01,   ,       ,       ,       +
+,      1,      1.0,    2,      0.3,    3,      0.09
DVPREL1,209,    PSHELL, 109,    4,      0.01,   ,       ,       ,       +
+,      1,      1.0,    2,      0.2,    3,      0.04
```

```
DVPREL1,210,    PSHELL, 110,    4,      0.01,   ,       ,       ,       +
+,      1,      1.0,    2,      0.1,    3,      0.01
```

Listing 3-1.

The three design variables, or basis function multipliers, are defined on the three DESVAR entries. Each has an initial value of 0.33; therefore, the initial design is defined as that which has a one-third contribution from each of the basis functions. The lower and upper bounds on the design variables are –1.0 and +1.0, respectively, which allow negative as well as positive contributions from each of the basis functions. The thicknesses are never allowed to assume negative values due to the 0.01 value of PMIN, or minimum allowable property value, in field 6 of each of the DVPREL1 entries.

Since the design model overrides the analysis model in design optimization, the initial analysis model properties will be set equal to the values computed from the design model specification. For this reason, no effort has been made to calculate the initial thickness distribution on the set of PSHELL entries. These thicknesses have arbitrarily been specified as 1.0 since they will just be overridden anyway.

Note

A table is output whenever a design model override of the analysis model occurs. A sample of this table is shown in Design Optimization Output.

Each DVPREL1 entry defines a row in the reduced basis matrix. The continuation lines of each entry lists the design variables and coefficients in each of the linear relations. Note that each of these continuation lines is simply a row in the reduced basis matrix.

Limitations in Reduced Basis Vector Formulations

In the previous example, the design variable-to-property relations were defined in terms of a reduced basis formulation. Essentially, the optimizer’s task is to find the best combination of these basis vectors. This combination might not yield the true optimum, and in fact only does so if some combination of the basis vectors is able to uniquely define the optimum. Including other basis vectors or allowing each property to vary independently probably will yield a different optimum, perhaps even one having a lower structural weight.

Which final design is more useful? If an arbitrary variation of plate thicknesses is permissible (assuming it can be manufactured, of course), and it did not require an unreasonable number of design variables to obtain this level of generality, then an independent type of relation may be preferable. That is, each plate element thickness can be a function of just a single design variable. However, in the preceding example, the basis function approach was able to guarantee that the resulting plate section had a smoothly varying thickness (in a stepwise fashion, that is). This may be a more desirable design solution, even though the objective function might be reduced further if this condition were relaxed. In short, the design model formulation represents a statement of design expectations and an implicit acknowledgement of its limitations.

Linking with User-Defined Equations

If the number of analysis model properties cannot be changed or reduced and a reduced basis formulation with DVPREL1 type relations is not sufficient or preferred, use of the DLINK (Design variable LINKing) Bulk Data entry should be considered. Its purpose is to provide explicit linear relations among the design variables themselves.

The DLINK entries define relations of the type

$$\{x\} = \begin{Bmatrix} x_I \\ x_D \end{Bmatrix} = \begin{Bmatrix} 0 \\ C_0 \end{Bmatrix} + \begin{bmatrix} I \\ C_1 \end{bmatrix} \{x_I\}$$

Equation 3-4.

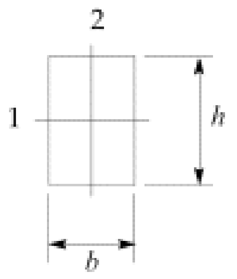
Where the items in braces are separately vectors, and the items in brackets (0 and C_1) are matrices. Eq. 3-4 indicates that design variable linking effectively partitions the vector of design variables into an independent set $\vec{x}_I$ and a dependent set $\vec{x}_D$. Each DLINK entry (each row) in Eq. 3-4 defines a single dependent variable as a linear combination of an independent set of design variables. The dependent design variable still needs to be defined using a DESVAR entry. The DLINK entry just describes the explicit linking relations.

A dependent design variable can be used in the design model in exactly the same way as an independent design variable. It can be used in connection with either property or shape optimization, on DVPREL1 and DVPREL2 relations, and so on. There are no restrictions. The optimizer, however, need only consider those variables that are in the independent set. The corresponding dependent design variable changes are determined from the independent set using the linking defined by Eq. 3-4.

Example

Quite often, the DLINK entry may be the only way to implement a given design variable linking scheme. For example, imagine that a rectangular cross-sectional bar consisting of ten elements is to be designed. This bar should be smoothly tapering (in a stepwise fashion, at least), and for purposes of this example, the shape functions of Figure 3-3, are used.

Suppose the base dimension of this rectangular section is to be a constant 5 mm with the section height as the design variable. This choice yields a total of ten variables similar to the thicknesses of the previous example. However, the only way to describe the cross-sectional properties in terms of the cross-sectional dimensions is through the use of equations. The figure below shows a representative cross section and element orientation, and the equations for area, I_1 and I_2. These equations have been evaluated for an initial h of 10 mm.



$$(A = b) \cdot h = 5.0\text{E-}5\ m^2$$

$$I_1 = \frac{bh^3}{12} = 4.167\text{E-}10\ m^4$$

$$I_2 = \frac{hb^3}{12} = 1.042\text{E-}10\ m^4$$

Since the design variable-to-property relations for I_1 and I_2 are not linear in the design variables, DVPREL1 relations cannot be used to express the reduced basis formulation as before. The only way this can be implemented is by explicitly linking together the design variables themselves together using DLINK entries. Three additional design variables are defined and used as basis function multipliers. These design variables form the independent set. The dependent design variables, linked to the independent set, define the element dimensions. The corresponding element properties are defined using these dependent variables, which are input to DEQATN entries via DVPREL2 entries.

The design model definition is given in Listing 3-2. The analysis model contains ten independent PBAR entries since all ten bar sections in the model are to vary.

```
$PBAR,   PID,    MID,    A,      I1,     I2
PBAR,    101,    100,    5.E-5,  4.167-10,1.042-10
PBAR,    102,    100,    5.E-5,  4.167-10,1.042-10
PBAR,    103,    100,    5.E-5,  4.167-10,1.042-10
PBAR,    104,    100,    5.E-5,  4.167-10,1.042-10
PBAR,    105,    100,    5.E-5,  4.167-10,1.042-10
PBAR,    106,    100,    5.E-5,  4.167-10,1.042-10
PBAR,    107,    100,    5.E-5,  4.167-10,1.042-10
PBAR,    108,    100,    5.E-5,  4.167-10,1.042-10
PBAR,    109,    100,    5.E-5,  4.167-10,1.042-10
PBAR,    110,    100,    5.E-5,  4.167-10,1.042-10
$
$...INDEPENDENT DESIGN VARIABLE SET:
$DESVAR,ID,      LABEL,  XINIT,  XLB,    XUB,    DELXV
DESVAR, 1,       X1,     1.0,    -1.0,   +1.0
DESVAR, 2,       X2,     0.0,    -1.0,   +1.0
DESVAR, 3,       X3,     0.0,    -1.0,   +1.0
$
$...DEPENDENT DESIGN VARIABLE SET:
$DESVAR,ID,      LABEL,  XINIT,  XLB,    XUB,    DELXV
DESVAR, 11,      H1,     .01,    1.E-4,  1.0
DESVAR, 12,      H2,     .01,    1.E-4,  1.0
DESVAR, 13,      H3,     .01,    1.E-4,  1.0
DESVAR, 14,      H4,     .01,    1.E-4,  1.0
DESVAR, 15,      H5,     .01,    1.E-4,  1.0
DESVAR, 16,      H6,     .01,    1.E-4,  1.0
DESVAR, 17,      H7,     .01,    1.E-4,  1.0
DESVAR, 18,      H8,     .01,    1.E-4,  1.0
DESVAR, 19,      H9,     .01,    1.E-4,  1.0
DESVAR, 20,      H10,    .01,    1.E-4,  1.0
$
$...DESIGN VARIABLE LINKING, Hj = Hj(X1,X2,X3) ; j=11,20:
$DLINK, ID,      DDVID,  CO,     CMULT,  IDV1,   C1,     IDV2,   C2,     +
$+,     IDV3,    C3,     ...
DLINK,  21,      11,     ,       .01,    1,      1.0,    2,      1.0,    +
+,      3,       1.0
DLINK,  22,      12,     ,       .01,    1,      1.0,    2,      0.9,    +
+,      3,       0.81
DLINK,  23,      13,     ,       .01,    1,      1.0,    2,      0.8,    +
+,      3,       0.64
DLINK,  24,      14,     ,       .01,    1,      1.0,    2,      0.7,    +
+,      3,       0.49
DLINK,  25,      15,     ,       .01,    1,      1.0,    2,      0.6,    +
+,      3,       0.36
DLINK,  26,      16,     ,       .01,    1,      1.0,    2,      0.5,    +
+,      3,       0.25
DLINK,  27,      17,     ,       .01,    1,      1.0,    2,      0.4,    +
+,      3,       0.16
DLINK,  28,      18,     ,       .01,    1,      1.0,    2,      0.3,    +

+,      3,       0.09
DLINK,  29,      19,     ,       .01,    1,      1.0,    2,      0.2,    +
+,      3,       0.04
DLINK,  30,      20,     ,       .01,    1,      1.0,    2,      0.1,    +
+,      3,       0.01
$
$...DESIGN EQUATION:
DEQATN  40       I2(H) = 0.005*H**3/12.
$
$...TYPE-2 DESIGN VARIABLE TO PROPERTY RELATIONS:
$DVPREL2,ID,     TYPE,   PID,    FID,    PMIN,   PMAX,   EQID,   ,       +
$+,     DESVAR, DVID1,  DVID2,  ...,    ,       ,       ,       ,       +
$+,     DTABLE, CID1,   CID2,   ...
```

```
DVPREL2,41,     PBAR,   101,    5,      1.0E-12,,       40,     ,       +
+,      DESVAR, 11
DVPREL2,42,     PBAR,   102,    5,      1.0E-12,,       40,     ,       +
+,      DESVAR, 12
DVPREL2,43,     PBAR,   103,    5,      1.0E-12,,       40,     ,       +
+,      DESVAR, 13
DVPREL2,44,     PBAR,   104,    5,      1.0E-12,,       40,     ,       +
+,      DESVAR, 14
DVPREL2,45,     PBAR,   105,    5,      1.0E-12,,       40,     ,       +
+,      DESVAR, 15
DVPREL2,46,     PBAR,   106,    5,      1.0E-12,,       40,     ,       +
+,      DESVAR, 16
DVPREL2,47,     PBAR,   107,    5,      1.0E-12,,       40,     ,       +
+,      DESVAR, 17
DVPREL2,48,     PBAR,   108,    5,      1.0E-12,,       40,     ,       +
+,      DESVAR, 18
DVPREL2,49,     PBAR,   109,    5,      1.0E-12,,       40,     ,       +
+,      DESVAR, 19
DVPREL2,50,     PBAR,   110,    5,      1.0E-12,,       40,     ,       +
+,      DESVAR, 20
```

Listing 3-2.

The basis function multipliers are x1, x2, and x3. Their initial values of 1.0, 0.0, and 0.0 correspond to a constant height beam section. Negative lower bounds allow a negative contribution from any of the basis functions.

The dependent variables are defined using DESVAR entries 11 through 20. Since the dependent variables define the bar element heights (i.e., nonnegative physical dimensions), their lower bounds are specified as small positive numbers to avoid negative bar cross-sectional heights.

Note

The optimizer never proposes a design that is outside the bounds placed on the design variables. (This often results in a more effective method of applying bounds than is PMIN or PMAX on the DVPRELi Bulk Data entries.)

The dependent variables are related to the independent set on DLINK entries 21 through 30. Each DLINK entry defines a row of the reduced basis matrix similar to the DVPREL1 entries of the previous example. This dependent set of design variables is input to equations in DVPREL2 type relations.

DVPREL2 entries 41 through 50 specify the area moments of inertia about axis 1 for each of the bar element properties as indicated by the 5 in field 5. Note that they all reference the same DEQATN entry 40 containing the functional relation between h and I_1. Each of the DVPREL2 continuations describe the equation input arguments, consisting of just a single design variable. The 1.0E–12 value for PMIN in field 6 ensures that the property does not take on values less than this bound.

For brevity, the equations for areas and (and DEVPREL2) for areas and I_2 for each property group have not been listed since the method used to define these relations is identical to that shown here. Without relating the design variables to changes in bar areas, weight minimization, for example, will not be possible. Also, if stresses due to bending are in the design model, the dependence of the stress recovery locations on the design variables should also be defined.

Constraint Screening

Constraint screening is an automatic process for both design sensitivity and optimization and consists of two parts: regionalization and deletion. You can override some of the defaults associated with this process by

- Providing explicit region IDs on the DRESP1 and DRESP2 entries.
- Using DSCREEN entries to override key screening parameters.

There is often little need to override the constraint screening defaults, since these perform well in the vast majority of cases. However, computational efficiency can sometimes be gained by modifying these defaults in special cases. This section outlines constraint screening so that you can make effective use of it should the occasion arise.

Basic Concepts

The basic assumption behind constraint screening is that most structural optimization problems contain more constraints than are necessary to adequately guide the design. For example, suppose bending stresses for every beam element are required to be less than some allowable upper limit. Not every element in the model may be critically loaded, while those that are may undergo more or less similar changes as the design is modified. This is a case of extraneous and redundant design data. If the constraints can be “filtered” in a consistent manner, a considerable increase in efficiency may be realized.

A response that is to be constrained must first be defined on either a DRESP1 or DRESP2 entry. Response lower and upper bounds are set with a DCONSTR entry. For the *j*-th response this definition yields

$$r_j^L \le r_j(\vec{x}) \le r_j^U$$

Equation 3-5.

NX Nastran uses these bounds to create a pair of normalized constraints as

$$g_{2j-1}(\vec{x}) = \frac{r_j^L - r_j(\vec{x})}{\left|r_j^L\right|} \le 0$$
$$g_{2j}(\vec{x}) = \frac{r_j(\vec{x}) - r_j^U}{\left|r_j^U\right|} \le 0$$

Equation 3-6.

where the absolute values of the bounds are used as normalizing factors.

This normalization provides a convenient method of screening the constraints. For example, any normalized constraint with a value of +0.5 has violated its bound by 50%, while a constraint with a value of –0.5 is within 50% of its bound. We might argue that it may not be necessary to retain every constraint, just those that are greater than some normalized value.

Constraint Deletion

Figure 3-4 represents a group of constraints in bar chart form. In constraint deletion, any constraint exceeding the truncation threshold value (shown as TRS in the figure) is retained for the current design cycle. Those constraints whose values are less than this threshold are temporarily deleted for the current cycle. These deleted constraints may again become active during subsequent design cycles and thus retained.

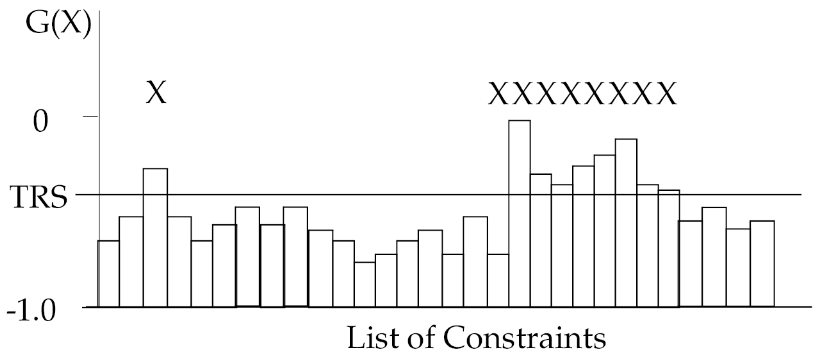


Figure 3-4. Constraint Deletion

The constraints retained for the current design cycle are denoted by an ‘X’ in the figure. You can change the value of TRS from its default of –0.5 using a DSCREEN entry. TRS is defined on a per response-type basis.

Constraint Regionalization

Each set of structural responses is automatically assigned a unique “region” identifier. All constraints associated with this region for a given subcase comprise a unique group. The assumption is that constraints from this group are likely to contain redundant information from this group, only a maximum number of constraints is retained (as determined by NSTR on the DSCREEN entry).

Note

Responses are usually grouped into regions based on their DRESP1 entry identification or by their property group association. See the DSCREEN entry for default region specifications.

For example, imagine a highly stressed, constant thickness panel comprised of a large number of shell elements. Suppose we are seeking to redesign the panel thickness, subject to element stress constraints. Due to the large loads, a number of elements in this panel may yield stress constraints that are well above the truncation threshold TRS. They will thus pass the first screening test. However, this still gives us more design information than we really need since, as the panel thickness is changed, the stresses will probably vary more or less in unison. It is probably reasonable to consider only the few largest stresses in this region and temporarily ignore the rest even though they are greater than TRS. Grouping similar constraints together is the idea behind constraint regionalization.

Note

The default for NSTR is 20. In a region of high stress, modeled using a fine mesh of plate elements, reduction to an NSTR number of constraints could easily lead to a hundredfold reduction in the number of design constraints.

Suppose the first 30 constraints in Figure 3-4 actually belong to three separate regions as shown in Figure 3-5. In Region 3, eight constraints are greater than or equal to the truncation threshold. It may be advantageous to consider only a few of the largest retained constraints from this region and disregard the others. Suppose that NSTR = 2 for Region 3. Then only two out of the eight possible constraints in this region are retained and six are discarded even though their values exceed the truncation threshold. In Regions 1 and 2, less than two constraints are retained; therefore, no further deletion takes place. In the end, only three constraints out of the original 30 are retained in the approximate problem.

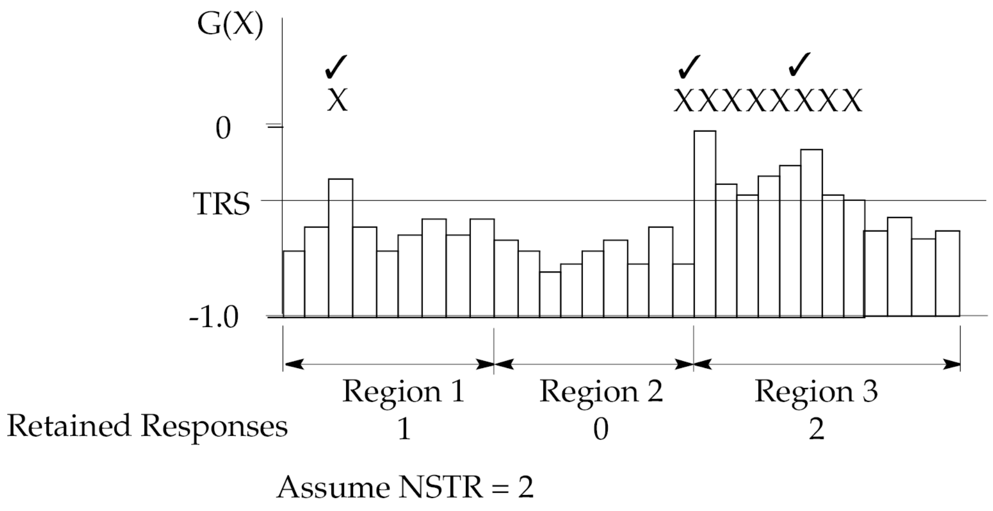


Figure 3-5. Constraint Regionalization

Constraint Screening and Sensitivity Analysis

Constraint screening precedes sensitivity analysis as can be seen from the flowchart in Figure 3-1 and in the detailed Solution 200 flowchart in Design Modeling for Sensitivity and Optimization. The default values of TRS and NSTR are usually fine for most design optimization problems; however, for design sensitivity analysis, these default values may not be satisfactory. It may be that some of the responses for which you had requested that sensitivities be computed are screened out. To ensure that all responses are retained, you may need to set TRS to a large negative number, perhaps –10. or –100. Also, depending on the number of responses per region, it may be necessary to increase NSTR. This combination ensures that all constraints pass both levels of screening and all responses are retained for sensitivity analysis. This procedure is the only way to turn off the screening.

Formal Approximations

NX Nastran uses formal approximation to avoid the high cost associated with repeated finite element analyses during design optimization. Two basic types of approximation are used

- Direct variable approximations.
- Reciprocal variable approximations.

These approximations are used in various ways according to three different methods available in NX Nastran:

- Direct Approximations (APRCOD = 1).
- Mixed Approximations (APRCOD = 2 default).
- Convex Linearization (APRCOD = 3).

Note

A DOPTPRM entry is used to select APRCOD.

Choosing to override the default approximation is usually problem-dependent, and often without strict rules or guidelines. This subsection presents theoretical details that might help you decide when such an override might be appropriate.

The formal approximations used in NX Nastran are all based on simple first-order Taylor series expansions. The general form of this expansion is

$$\tilde{r}_j\left(\vec{x}^0 + \Delta\vec{x}\right) = r_j(\vec{x}^0) + \sum_i \left.\frac{\partial r_j}{\partial x_i}\right|_{\vec{x}^0} \cdot \Delta x_i$$

Equation 3-7.

Approximation Errors

The required derivative information is available from the design sensitivity analysis as outlined in Design Sensitivity Analysis. Note that the approximate response, $\tilde{r}_j(\vec{x}^0 + \Delta\vec{x})$, in Eq. 3-7 is linear in the design variable change Δ_x. Thus, we expect some error when we try to approximate responses that are actually nonlinear in Δ_x. Figure 1-16 in Structural Optimization shows how the error is a function of Δ_x.

Move Limits and Approximations

During approximate optimization, the code places limits on the maximum allowable moves in the design space in order to minimize the errors associated with these approximations. In NX Nastran, move limits are placed both on property as well as design variable changes. Move limits are discussed in Optimization with Respect to Approximate Models.

Direct Approximations

A direct variable approximation linearizes the function directly in terms of the design variables. For the *j*-th constraint function, the direct approximation is written as

$$\tilde{g}_{jD}\left(\vec{x}^{0} + \Delta\vec{x}\right) = g_j(\vec{x}^{0}) + \sum_i \left.\frac{\partial g_j}{\partial x_i}\right|_{\vec{x}^{0}} \cdot \Delta x_i$$

Equation 3-8.

where the subscript *D* indicates a direct approximation.

The quantity $\Delta\vec{x}$ represents a total vector move in the design space from the initial design $\vec{x}^{0}$. The partial derivatives $\partial g_j/\partial x_i$ are available from the design sensitivity analysis. An equation similar to Eq.3-8 can be written for the objective function, where *F* replaces *g* in the expression.

Reciprocal Approximations

The first-order Taylor series expansions of Eq. 3-8 can alternately be expressed in terms of reciprocal variables. This choice turns out to be quite useful for those responses that are inversely proportional to the design variables. This simple idea can be easily shown in the case of the axially loaded rod element of Figure 3-6 where the cross-sectional area A is taken as the design variable.

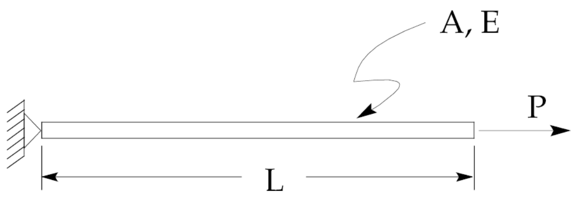


Figure 3-6. Axially Loaded Bar Element

The axial stress in the bar is equal to *P/A*, and the displacement is equal to *PL/AE*. If these responses are used in the design model, linearizing with respect to the quantity *1/A* will produce approximations that are exact in both cases. In the general case, of course, these approximations are not exact due to the static indeterminacy of the structure. However, for all element types, the proportionality of the stiffness matrices to the inverse of the sizing quantities forms a reasonable basis for arguing the use of reciprocal approximations.

Reciprocal approximations can be derived from linear approximations if we first substitute the intermediate variables y_i. For the approximate constraint of Eq. 3-8, we get

$$\tilde{g}_j\left(\vec{y}^{0} + \Delta\vec{y}\right) = g_j(\vec{y}^{0}) + \sum_i \left.\frac{\partial g_j}{\partial y_i}\right|_{\vec{y}^{0}} \cdot \Delta y_i$$

Equation 3-9.

Setting the intermediate variables to reciprocals of the design variables

$$y_i = \frac{1}{x_i}$$

Equation 3-10.

and noting that ∂y = –(1/x^2)∂x, and that ▵y = –▵x/(x°x), we obtain:

$$\tilde{g}_{jR}\left(\vec{x}^0 + \Delta\vec{x}\right) = g_j(\vec{x}^0) + \sum_i \left.\frac{\partial g_j}{\partial x_i}\right|_{\vec{x}^0} \cdot \frac{x_i^0}{x_i} \cdot \Delta x_i$$

Equation 3-11.

where the subscript *R* indicates a reciprocal approximation.

Approximation Methods

Three different approximation methods are available in NX Nastran. You can select any one of them using the APRCOD field on the DOPTPRM entry. These methods are as follows:

- APRCOD = 1: This specifies that direct approximations Eq. 3-8 are to be used for the objective function as well as all constraints. The resulting approximate optimization is then performed with respect to an entirely linear approximate design space. You should use this option if you know that the structural responses are well-approximated by linear functions in the design variables. Otherwise, without carefully chosen move limits, a greater number of approximate optimization cycles may be required before convergence is achieved.

- APRCOD = 2: (Default) This choice specifies that mixed approximations (or more precisely, a mixture of approximations) are to be used. Direct approximations are used for volume, weight, element force, and buckling load responses, while reciprocal approximations are used for all other response types. This method works well in a wide variety of problem types. Because of its reliability, it has been selected as the default method.

- APRCOD = 3: This selects the convex linearization method. Essentially, this method chooses either a direct or reciprocal constraint approximation depending on which one provides the larger estimation of the constraint function. In other words, this method chooses the more conservative of the two approximations.

  For example, the direct approximation for the *j*-th constraint is

$$\tilde{g}_{jD}\left(\vec{x}^0 + \Delta\vec{x}\right) = g_j\left(\vec{x}^0\right) + \sum_i \left.\frac{\partial g_j}{\partial x_i}\right|_{\vec{x}^0} \cdot \Delta x_i$$

Equation 3-12.

while the reciprocal approximation is

$$\tilde{g}_{jR}\left(\vec{x}^{0} + \Delta\vec{x}\right) = g_j\left(\vec{x}^{0}\right) + \sum_i \left.\frac{\partial g_j}{\partial x_i}\right|_{\vec{x}^{0}} \cdot \frac{x_i^0}{x_i} \cdot \Delta x_i$$

Equation 3-13.

Since both approximations are readily available (both use the same gradient information), the choice of which to use can be made just by looking at the sign of the difference between the two:

$$\tilde{g}_{jD}\left(\vec{x}^{0} + \Delta\vec{x}\right) - \tilde{g}_{jR}\left(\vec{x}^{0} + \Delta\vec{x}\right) = \sum_i \left.\frac{\partial g_j}{\partial x_i}\right|_{\vec{x}^{0}} \frac{(x_i - x_i^0)^2}{x_i}$$

Equation 3-14.

Since the squared term in the expression is always positive, the choice depends on the sign of the product:

- if $\left.\frac{\partial g_j}{\partial x_i}\right|_{\vec{x}^{0}} \frac{1}{x_i} \geq 0$ Use the direct approximation for x_i since it yields a larger estimate of constraint value.

Equation 3-15.

- if $\left.\frac{\partial g_j}{\partial x_i}\right|_{\vec{x}^{0}} \frac{1}{x_i} < 0$ Use the reciprocal approximation for x_i since it yields a larger estimate of constraint value.

Equation 3-16.

This criterion is applied on an individual design variable basis. Thus, a combination of direct and reciprocal approximations can be made for a single constraint.

For convex linearization, the objective function is always linearized, regardless of its response type.

Modeling Guidelines

There are several situations where the default mixed method (APRCOD=2) is not applicable and therefore not recommended. The first is when basis vectors for either property or shape optimization are used. In this case, the physical justification for the use of reciprocal approximations does not apply (see Reciprocal Approximations earlier in this chapter), and either direct approximations (or perhaps even convex linearization) should be used instead.

The second situation occurs if design variables can pass through zero during the course of optimization. This can occur if the lower bound on the DESVAR entry is a negative quantity and the upper bound is positive. In this case, APRCOD is automatically set to 1 to avoid ill-conditioning about the zero value of the design variable.

Finally, experience has also shown that the mixed method is not always the best performer for dynamic response optimization tasks. For these highly nonlinear responses, direct approximations, coupled with more stringent move limits are often more reliable.

Advanced Topics in Approximations

The direct and reciprocally-approximated constraints of Eq. 3-8 and Eq. 3-11 require the evaluation of function derivatives with respect to design variables. Computing these function derivatives is complicated somewhat by design variable-to-property relations, which may be either type-1 or type-2 (DVPREL1 or DVPREL2), and responses, which may be either first- or second-level (DRESP1, DRESP2, or DRESP3).

To accommodate type-1 and type-2 design variable-to-property relations, the concept of a design variable is generalized internally in the code. For type-1 relations, the basis for the constraint and objective function approximations is the set of independent design variables. For type-2 relations, this basis is the set of properties defined on the DVPREL2 entries. For the *j*-th constraint, these choices result in

$$\tilde{g}_j\left(\vec{x}^0 + \Delta\vec{x}\right) = g_j\left(\vec{x}^0\right) + \sum_i \left.\frac{\partial g_j}{\partial x_i}\right|_{\vec{x}^0} \cdot \Delta x_i \qquad \text{(DVPREL1)}$$

Equation 3-17.

$$\tilde{g}_j\left(\vec{x}^0 + \Delta\vec{x}\right) = g_j\left(\vec{x}^0\right) + \sum_i \left.\frac{\partial g_j}{\partial p_i}\right|_{\vec{x}^0} \cdot \Delta p_i \qquad \text{(DVPREL2)}$$

Equation 3-18.

where:

$$\Delta p_i = p_i\left(\vec{x}^0 + \Delta\vec{x}\right) - p_i\left(\vec{x}^0\right)$$

Equation 3-19.

Similar expressions can also be written for the objective function. The benefit of this generalization is that properties are computed precisely in Eq. 3-19 for a given change in design variables. The approximation in Eq. 3-18 thus retains all of the non-linearities of the DVPREL2 relations, yielding a more accurate approximation.

Depending on whether the response quantities are first- or second-level, the derivatives are evaluated as follows. If we let the quantity *f* represent either an objective or a constraint function, then for first-level responses we have

$$\frac{\partial f}{\partial x_i} = \sum_j \frac{\partial f}{\partial r_j^{(1)}} \cdot \frac{\partial r_j^{(1)}}{\partial x_i} \qquad \text{(DVPREL1)}$$

Equation 3-20.

$$\frac{\partial f}{\partial p_i} = \sum_j \frac{\partial f}{\partial r_j^{(1)}} \cdot \frac{\partial r_j^{(1)}}{\partial p_i} \qquad \text{(DVPREL2)}$$

Equation 3-21.

where the superscript on $r^{(1)}$ denotes a first-level (DRESP1) response.

For objectives or constraints that are functions of second-level responses, we have

$$\frac{\partial f}{\partial x_i} = \sum_j \frac{\partial f}{\partial r_j^{(2)}} \cdot \frac{\partial r_j^{(2)}}{\partial x_i} \qquad \text{(DVPREL1)}$$

Equation 3-22.

$$\frac{\partial f}{\partial p_i} = \sum_j \frac{\partial f}{\partial r_j^{(2)}} \cdot \frac{\partial r_j^{(2)}}{\partial p_i} \qquad \text{(DVPREL2)}$$

Equation 3-23.

where $r^{(2)}$ is a second-level response.

In general, these second-level responses can be functions of design variables, table constants, first-level responses, and grid coordinates. That is,

$$r_j^{(2)} = r_j^{(2)}(x, c, r^{(1)}, \{G\})$$

Equation 3-24.

Chain rule differentiation of Eq. 3-24 with respect to the *i*-th design variable gives (using ∂ to denote partial differentiation with respect to the design variables appearing explicitly in r_j)

$$\frac{\partial r_j^{(2)}}{\partial x_i} = \frac{\delta r_j^{(2)}}{\delta x_i} + \sum_k \frac{\partial r_j^{(2)}}{\partial r_k^{(1)}} \cdot \frac{\partial r_k^{(1)}}{\partial x_i} + \sum_l \frac{\partial r_j^{(2)}}{\partial \{G\}_l} \cdot \frac{\partial \{G\}_l}{\partial x_i}$$

Equation 3-25.

where x_i is used here to indicate either a design variable or a “design property” (DVPREL2 property), see Eq. 3-22 or Eq. 3-23.

In NX Nastran, the terms in Eq. 3-20 through Eq. 3-25 are computed as follows:

1. $\frac{\partial r_j^{(1)}}{\partial x_i}, \frac{\partial r_j^{(1)}}{\partial p_i}$ These quantities are available directly from the semi-analytic sensitivity analysis.

2. $\frac{\partial f}{\partial r_j^{(1)}}, \frac{\partial f}{\partial r_j^{(2)}}$ Depending on whether is an objective or constraint, these terms are available as

$$\text{objective: } \frac{\partial f}{\partial r_j^{(1)}} = \frac{\partial f}{\partial r_j^{(2)}} = 1$$

Equation 3-26.

$$\text{constraint: } \frac{\partial f}{\partial r_j^{(1)}} = \frac{\partial f}{\partial r_j^{(2)}} = -\frac{1}{\left|r_j^L\right|} \text{ lower-bound constraint}$$

$$\frac{\partial f}{\partial r_j^{(1)}} = \frac{\partial f}{\partial r_j^{(2)}} = -\frac{1}{\left|r_j^U\right|} \text{ upper-bound constraint}$$

Equation 3-27.

3. $\frac{\delta r_j^{(2)}}{\delta x_i}$ This term is approximated using central differences and the equation solving utility in NX Nastran as

$$\frac{\delta r_j^{(2)}}{\delta x_i} = \frac{\Delta r_j^{(2)}}{\Delta x_i} \simeq \frac{r_j^{(2)}\left(\vec{x}^0 + \Delta x_i\right) - r_j^{(2)}\left(\vec{x}^0 - \Delta x_i\right)}{2\Delta x_i}$$

Equation 3-28.

where Δx_i = FDCH · x_i^0 (with FDCH = 1.0E-3 currently in the program) and if FDCH · x_l^0 < FDCHM (with FDCHM = 1.0E-6 currently in the program)

4. $\dfrac{\partial r_j^{(2)}}{\partial r_k^{(1)}}$ This term is also approximated using central differences and the equation utility as

$$\frac{\partial r_j^{(2)}}{\partial r_k^{(1)}} \cong \frac{\Delta r_j^{(2)}}{\Delta r_k^{(1)}} = \frac{r_j^{(2)}\left(\overrightarrow{r^{(1)}} + \Delta r_k^{(1)}\right) - r_j^{(2)}\left(\overrightarrow{r^{(1)}} - \Delta r_k^{(1)}\right)}{2\Delta r_k^{(1)}}$$

Equation 3-29.

where $\Delta r_k^{(1)}$ is determined using FDCH and FDCHM as in item 3.

5. $\dfrac{\partial r_j^{(2)}}{\partial \{G\}_l}$

This term is approximated using central differences as

$$\frac{\partial r_j^{(2)}}{\partial \{G\}_l} \cong \frac{\Delta r_j^{(2)}}{\{\Delta G\}_l} = \frac{r_j^{(2)}(\{G^0\} + \{\Delta G\}_l) - r_j^{(2)}(\{G^0\} - \{\Delta G\}_l)}{\{\Delta G\}_l}$$

Equation 3-30.

where $\{\Delta G\}_l$ is the variation in the coordinates of the *l*th grid point and $\{G^0\}$ represents the baseline coordinates of those grids used in the expression for $r_j^{(2)}$.

$\{\Delta G\}_l$ is determined as

$$\{\Delta G\}_l = \text{FDCH} \cdot \{G^0\}_l$$

Equation 3-31.

where *FDCHM* is used for any component that is less than this amount.

6. $\dfrac{\partial \{G\}_l}{\partial x_i}$

This information is available directly from the basis vectors for shape optimization. Note that each term of a basis vector defines the (constant) rate of change of a grid coordinate given a change in a design variable.

3.2 Design Sensitivity Analysis

Design sensitivity analysis computes the rates of change of structural response quantities with respect to changes in design variables. Since these partial derivatives provide the essential gradient information to the optimizer, they are always computed in connection with design optimization. Sensitivity analysis is always performed automatically in NX Nastran whenever design optimization is requested. There may be times when you want the software to compute just the sensitivity coefficients and not perform optimization. You can then use this sensitivity information to perform parametric design studies or to link with your own optimizer.

This section presents the theoretical basis for design sensitivity analysis in NX Nastran. It shows how to request sensitivity analysis, describes how the sensitivity coefficients are computed, and presents modeling guidelines.

General Considerations

A design sensitivity coefficient is defined as the rate of change of a particular response quantity *r* with respect to a change in a design variable *x* or, in other words, as $\partial r/\partial x$. These coefficients are evaluated at a particular design characterized by the vector of design variables $\vec{x}^0$, giving

$$\lambda_{ij} = \left.\frac{\partial r_j}{\partial x_i}\right|_{\vec{x}^0}$$

Equation 3-32.

where subscripts are used to indicate the *i*-th design variable and the *j*-th response. Eq. 3-32 is just the slope of the response with respect to x_i as is shown in Eq. 3-7.

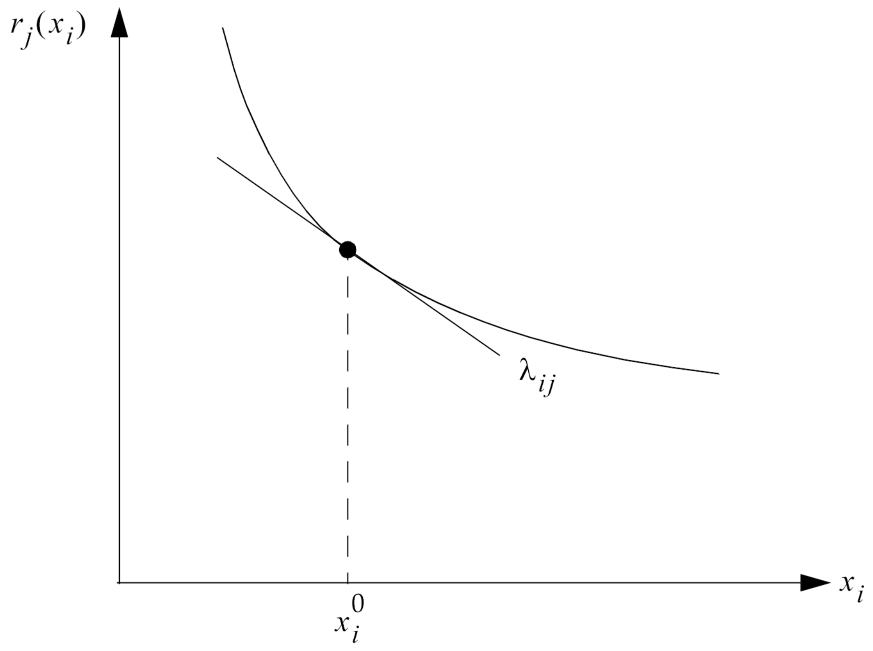


Figure 3-7. Design Sensitivity Coefficient—Graphical Interpretation

For Which Responses Are Sensitivities Computed?

Sensitivities are computed for all of the responses (both first- and second-level) that are used to define the objective function and retained constraints.

Recall from Approximation Concepts in Design Optimization that sensitivity analysis is always performed after the constraint screening phases are completed (Figure 3-1). Thus, fewer response sensitivities may be computed than there are responses defined in the design model. Of course, this is the goal since the computational effort associated with sensitivity analysis is reduced. (If you wish to force the computation of all sensitivities for all responses defined in the design model, you can effectively disable the screening process by using a large negative TRS on the DSCREEN Bulk Data entries along with a large NSTR if necessary. See the DSCREEN entry description in Input Data for further details.)

Requesting Sensitivity Analysis Output

Design sensitivities are output in the form of Eq. 3-1. In Solution 200, you request design sensitivity analysis and output with either the DSAPRT Case control command or the following Bulk Data parameter definition:

```
PARAM, OPTEXIT, ±4
```

Note

You can also request design sensitivity coefficient output with an OPTEXIT value of 7. See Solution 200 Program Flow), Solution 200 Program Flow, for details.

An OPTEXIT of ±4 prints the sensitivity coefficients to the standard output file. A value of –4 outputs the coefficients in binary form using OUTPUT4 formats. DSAPRT has priority over the PARAM, OPTEXIT, ±4 (or 7) option. It does have somewhat more flexibility and options, such as formatting.

Identifying the Sensitivity Coefficients

If requested by a PARAM,OPTEXIT,±4 or 7, or by the “unformatted” DSAPRT Case Control command, sensitivity coefficients are output in the matrix DSCM2. If you request sensitivity coefficients with the formatted DSAPRT Case Control command, then the software prints the coefficients in a table with headers. You can use both options with an EXPORT option to write the sensitivities to an external binary file.

Each column in the matrix DSCM2 corresponds to a particular response quantity, while each row corresponds to a design variable. The output is formatted in one column after another, in sequence, such as response by response. The component of the *i*-th row and *j*-th column is

$$(\mathrm{DSCM2})_{ij} = \left.\frac{\partial r_j}{\partial x_i}\right|_{\vec{x}^0}$$

Equation 3-33.

A correlation matrix is also output to help identify the column order of the individual design responses. The row order in DSCM2 corresponds to the design variables defined on the set of DESVAR entries sorted by increasing design variable ID. See Output Features and Interpretation, Design Sensitivity Output, for examples illustrating the interpretation of this output.

Internal Representation of the Sensitivity Coefficients in NX Nastran

For reasons of efficiency as well as accuracy, NX Nastran uses a slightly different internal representation of these coefficients than described above, depending on how the design variables are related to the analysis model properties.

For DVPREL1 type relations, the set of independent design variables is taken as the basis for the design sensitivity coefficients (see the DLINK entry for a definition of independent and dependent design variables). This is an efficient choice since a large number of properties may be a function of a much smaller set of design variables.

For DVPREL2 relations, the basis for sensitivity analysis is taken to be the set of analysis model properties referenced on all of these second-level relations. Inside the code, the sensitivities are then

$$\lambda_{ij} = \left.\frac{\partial r_j}{\partial p_i}\right|_{\vec{x}^0}$$

Equation 3-34.

Choosing properties as the basis greatly improves the accuracy of the approximate responses used in design optimization. Since an explicit relation between the properties and the associated design variables is given by the DVPREL2–DEQATN entry pair, this information can be used to evaluate the properties for a given change in the set of design variables. Computing the sensitivities with respect to the design variables would have linearized this relation, resulting in a less accurate approximation.

To summarize, the sensitivity coefficients used internally in NX Nastran are

$$\text{DVPREL1:} \quad \frac{\partial r_j}{\partial x_i} \qquad i = 1, \text{NIDV}$$

$$\text{DVPREL2:} \quad \frac{\partial r_j}{\partial p_k} \qquad k = 1, \text{NIPR}$$

Equation 3-35.

where *NIDV* is the number of independent design variables and *NIPR* is the set of properties defined on all DVPREL2 entries.

To illustrate the advantage in using a mixed design variable-property basis, consider the normalized lower-bound constraint in Eq. 2-35, Defining the Constraints.

$$g_{2j-1}(\vec{x}) = \frac{r_j^L - r_j(\vec{x})}{\left|r_j^L\right|} \le 0$$

Equation 3-36.

This is the form of the lower-bound constraint that is automatically generated in NX Nastran from the DCONSTR entry data. For a finite move in the design space Δ*x*, the approximated constraint is

$$\tilde{g}_{2j-1}\left(\vec{x}^0 + \Delta\vec{x}\right) = \frac{r_j^L - \tilde{r}_j(\vec{x}^0 + \Delta\vec{x})}{\left|r_j^L\right|} \le 0$$

Equation 3-37.

Since the response is only approximate, the constraint is approximate as well. For responses that depend on properties that are linear functions of the design variables (DVPREL1), the approximation is

$$\tilde{r}_j(\vec{x}^0 + \Delta\vec{x}) = r_j(\vec{x}^0) + \sum_i \left.\frac{\partial r_j}{\partial x_i}\right|_{\vec{x}^0} \cdot \Delta x_i$$

Equation 3-38.

while, for responses that depend on the properties that are nonlinear functions of the design variables (DVPREL2),

$$\tilde{r}_j(\vec{x}^0 + \Delta\vec{x}) = r_j(\vec{x}^0) + \sum_k \left.\frac{\partial r_j}{\partial p_k}\right|_{\vec{x}^0} \cdot (p_k(\vec{x}^0 + \Delta\vec{x}) - p_k(\vec{x}^0))$$

Equation 3-39.

where the properties at the perturbed design in Eq. 3-39 can be evaluated precisely from the input equations DEQATN. Eq. 3-38 and Eq. 3-39 show the advantage of generalizing the basis to include not only design variables but also design properties.

Static Response Sensitivities

Requesting Static Response Sensitivities

To compute the sensitivities of static responses, you need to

- Define a design model (see Design Modeling for Sensitivity and Optimization).
- Define a static analysis subcase in Solution 200 using the Case Control command

```
ANALYSIS = STATICS
```

- Either request sensitivity coefficient output with the DSAPRT Case Control command or the Bulk Data parameter assignment

```
PARAM, OPTEXIT, 4
or
PARAM, OPTEXIT, 7
```

- Or, invoke Solution 200 for design optimization. (Static response sensitivities are computed automatically in design optimization if static responses are present in the design model.)

Theory

For a given structure with specified geometry, material properties, and boundary conditions, a displacement-based linear static analysis computes the displacement responses due to the applied loads. All other static responses such as stresses and strains are determined from the displacement solution. For an arbitrary response *r,* the functional dependency on displacement is written as

$$r = r(\vec{u})$$

Equation 3-40.

In the design context, the displacement solution is an implicit function of the design variables. That is,

$$\vec{u} = \vec{u}(\vec{x})$$

Equation 3-41.

Since all other design responses may be functions of these displacements, these are implicit in the design variables as well. These can include both first- as well as second-level response types where

$$\vec{r} = \vec{r}[\vec{u}(\vec{x})]$$

Equation 3-42.

In NX Nastran, the sensitivities of these responses with respect to changes in the design variables are approximated using first forward finite differences as

$$\frac{\partial r_j}{\partial x_i} \cong \frac{\Delta r_j}{\Delta x_i} = \frac{r_j(\vec{x}^0 + \Delta x_i, \vec{u}^0 + \Delta\vec{u}) - r_j(\vec{x}^0, \vec{u}^0)}{\Delta x_i}$$

Equation 3-43.

where r_j is the j-th design response.

Eq. 3-43 accounts for the functional dependency of the response on both the design variables and the displacement solution. However, since the displacements are implicit functions of the design variables, the quantity Δu must be computed as

$$\Delta\vec{u} = \frac{\partial\vec{u}}{\partial x_i} \cdot \Delta x_i$$

Equation 3-44.

Eq. 3-44 requires that the displacement sensitivity be known. The sensitivities can be computed by differentiating the equations of static equilibrium

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

Equation 3-45.

to obtain

$$[K]\frac{\partial\{u\}}{\partial x_i} = \frac{\partial\{P\}}{\partial x_i} - \frac{\partial[K]}{\partial x_i}\{u\}$$

Equation 3-46.

This equation is solved in NX Nastran to obtain the displacement sensitivities. Once these are known, the sensitivities of all necessary responses can be found from Eq. 3-43 and Eq. 3-44.

Computationally, the solution of Eq. 3-46 is identical to the solution for Eq. 3-45 except that they differ in the right-hand side load vectors. In fact, the cost of the displacement sensitivity solution can be reduced if the stiffness matrix, already available in decomposed form as a result of the finite element solution, is used. This observation is taken advantage of in NX Nastran.

Note that the right-hand side of Eq. 3-46 (often referred to as the pseudo-load vector), includes the partial derivatives $\partial\{P\}/\partial x_i$ and $\partial[K]/\partial x_i$. Since these are generally implicit functions of the design variables, these derivatives are approximated using finite differences as follows:

- For property optimization, unless CDIF = 'YES',

$$\frac{\partial[K]}{\partial x_i} \cong \frac{[K(\vec{x}^0 + \Delta x_i)] - [K(\vec{x}^0)]}{\Delta x_i}$$

$$\frac{\partial\{P\}}{\partial x_i} \cong \frac{\left\{P(\vec{x}^0 + \Delta x_i)\right\} - \left\{P(\vec{x}^0)\right\}}{\Delta x_i}$$

Equation 3-47.

- For shape optimization, or if CDIF = 'YES' (i.e. central finite differencing)

$$\frac{\partial[K]}{\partial x_i} \cong \frac{[K(\vec{x}^0 + \Delta x_i)] - [K(\vec{x}^0 - \Delta x_i)]}{2\Delta x_i}$$

$$\frac{\partial\{P\}}{\partial x_i} \cong \frac{\left\{P(\vec{x}^0 + \Delta x_i)\right\} - \left\{P(\vec{x}^0 - \Delta x_i)\right\}}{2\Delta x_i}$$

Equation 3-48.

For property optimization, the size of the move Δ*x* is given by

$$\Delta x_i = \mathrm{DELB} \bullet x_i$$

Equation 3-49.

where DELB is given on the DOPTPRM Bulk Data entry. See Shape Sensitivity (later in this chapter) for the corresponding grid variation for shape optimization.

As a result of these finite difference approximated terms in the pseudo-load vector, the solution of Eq. 3-46 is frequently referred to as a semianalytic one. A similar approach is used in the other analysis disciplines and is described in the following sections.

Even though Eq. 3-43 through Eq. 3-49 are expressed in terms of the design variables x_i , the independently varying properties p_i may also be used if DVPREL2 relations appear in the design model. In practice then, the set of design variables in the preceding equations is generalized in the code to include not only the independent design variables but also the independently varying properties.

Use of Adjoint Loads in Design Sensitivity Analysis

Design sensitivities can be calculated by solving Equation 3-46. This approach to calculating design sensitivities is often referred to as the pseudo-load method, where the right-hand side of Equation 3-46 represents the so-called pseudo-load.

The pseudo-load method is useful for calculating design sensitivities in many instances. However, the pseudo-load method may become cumbersome when:

- There are many design variables as compared to the number of response constraints for which gradients need to be calculated.
- The responses in question are functions of more than one of the basic responses such as displacement and velocity.

For these cases, NX Nastran automatically uses the adjoint load method.

- For the case of comparatively many design variables, adjoint loads are generated based on the number of responses.
- For the case where the responses are functions of more than one of the basic responses, basic adjoint loads can be combined to produce a composite adjoint load that leads to the gradient of the response.

The adjoint load for a response quantity is given as the derivative of that response quantity with respect to the vector of displacements, $\{u\}$. If the response quantity is denoted r_j, then the adjoint load for r_j is given by:

$$\partial r_j / \partial\{u\}$$

Equation 3-50.

As an example, suppose the response quantity is the static displacement of a degree-of-freedom at a grid point. That is, suppose $r_j = u_j$. Then the adjoint load is:

- 1.0 for that degree-of-freedom because $\partial u_j / \partial u_j = 1$.
- 0.0 for all other degrees-of-freedom because $\partial u_j / \partial u_i = 0$.

If r_j represents the static displacement of grid point k at an angle of +30° to a local X-axis in a two-dimensional displacement field, the adjoint loads that correspond to the static displacements in the X and Y-directions can be combined into a single adjoint load vector that corresponds to r_j as follows:

$$\partial r_j / \partial u_x + \partial r_j / \partial u_y = \cos(30°)\ \{i\} + \sin(30°)\ \{j\}$$

Equation 3-51.

where u_x and u_y are the static displacements at grid point k, and {i} and {j} are unit vectors in the X and Y-directions, respectively.

Application of an adjoint load on a finite element model yields a set of adjoint variables, $\{a\}$. For linear static behavior, the governing equation can be represented by:

$$[K]\ \{a\} = \partial r_j / \partial\{u\}$$

Equation 3-52.

where $[K]$ is the stiffness matrix of the finite element model.

The required sensitivity of the response r_j with respect to a particular design variable x_i is then given by:

$$dr_j / dx_i = \partial r_j / \partial x_i + \{a\}^T \{d\{p\} / dx_i - (d[K] / dx_i)\ \{u\}\ \}$$

Equation 3-53.

where:

$\{p\}$	Applied load on the finite element model
∂	Indicates “local” partial derivative
d	Indicates partial derivative over the system

As an example of a local derivative for a response quantity, consider the case where a stress component, σ_c, is the response quantity. The local derivative is given by:

$$\partial\sigma_c / \partial x_i = (\partial\{s\}^T / \partial x_i)\ \{u\}_e$$

Equation 3-54.

where $\{s\}^T$ is the relevant row of the transformation matrix from the element displacements, $\{u\}_e$, to the Gauss point stresses. Thus, it is defined by:

$$\sigma_c = \{s\}^T\ \{u\}_e$$

Equation 3-55.

Note

NX Nastran does not currently use adjoint loads for stress responses.

Where appropriate, the adjoint load method is also used for dynamic response sensitivities as an alternative to the pseudo-load method.

Eigenvalue Response Sensitivities

Requesting Eigenvalue Response Sensitivities

To compute eigenvalue sensitivities, you need to

- Define a design model (see Design Modeling for Sensitivity and Optimization).
- Define a normal modes subcase using the Case Control command

```
ANALYSIS = MODES
```

Eigenvalue response sensitivities are computed automatically in connection with design optimization. If you do not want to perform optimization, you can request sensitivity analysis with the DSAPRT Case Control command or the Bulk Data parameter assignment.

```
PARAM, OPTEXIT, 4
```

Theory

Normal modes analysis can be requested in Solution 200 by setting the Case Control command, ANALYSIS = MODES. This setting invokes both a normal modes analysis as well as a sensitivity analysis if either sensitivity or optimization is requested.

The eigenvalue equation is

$$([K] - \lambda_n [M])\{\phi_n\} = \{0\}$$

Equation 3-56.

where λ_n and φ_n are the *n*-th eigenvalue and eigenvector, respectively. [*K*] is the structural stiffness, [*M*] is the structural mass, and {*0*} is the null vector.

The governing Eq. 3-56 can be differentiated with respect to the *i*-th design variable x_i to yield,

$$([K] - \lambda_n [M])\frac{\partial\{\phi_n\}}{\partial x_i} + \left(\frac{\partial[K]}{\partial x_i} - \lambda_n \frac{\partial[M]}{\partial x_i}\right)\{\phi_n\} = \frac{\partial \lambda_n}{\partial x_i}[M]\{\phi_n\}$$

Equation 3-57.

When Eq. 3-57 is premultiplied by $\{\varphi_n{}^T\}$, the first term is zero. Eq. 3-57 can then be solved for the eigenvalue derivatives

$$\frac{\partial \lambda_n}{\partial x_i} = \frac{\{\phi_n\}^T\left(\frac{\partial[K]}{\partial x_i} - \lambda_n \frac{\partial[M]}{\partial x_i}\right)\{\phi_n\}}{\{\phi_n\}^T[M]\{\phi_n\}}$$

Equation 3-58.

In practice, the solution to the above equation is based on a semi-analytic approach. The derivatives of the mass and stiffness matrices are approximated using finite differences of the form in Eq. 3-47 and Eq. 3-48. Eq. 3-58 is solved for each retained eigenvalue referenced in the design model and for each design variable. This equation is valid only for the case of distinct eigenvalues.

Buckling Load Factor Sensitivities

Requesting Buckling Load Factor Sensitivities

To obtain buckling load factor sensitivities in Solution 200, you need to:

- Define a design model (see Design Modeling for Sensitivity and Optimization).
- Define a buckling subcase in Solution 200 with the Case Control command

```
ANALYSIS = BUCK
```

Buckling load factor sensitivities are computed automatically in connection with design optimization. If you just want to compute and output the sensitivity coefficients, you may do so with the DSAPRT Case Control command or the Bulk Data parameter assignment

```
PARAM, OPTEXIT, 4
```

Theory

The derivation of buckling sensitivity similar is similar to that of eigenvalue sensitivities. The equation for determining of the buckling load factor λ is

$$[K]\{\phi_n\} + \lambda_n[K_d]\{\phi_n\} = \{0\}$$

Equation 3-59.

where [K$_d$] is the differential stiffness (also known as the geometric stiffness) matrix.

Differentiating this expression and solving for the buckling load factor sensitivities yields

$$\frac{\partial\lambda_n}{\partial x_i} = \frac{\{\phi_n\}^T\left(\frac{\partial[K]}{\partial x_i} + \lambda_n\frac{\partial[K_d]}{\partial x_i}\right)\{\phi_n\}}{\{\phi_n\}^T[K_d]\{\phi_n\}}$$

Equation 3-60.

As before, this equation is solved semi-analytically by approximating the structural property matrix derivatives using finite differences.

Differential Stiffness Approximations

The approximation of the differential stiffness is worth discussing. This stiffness is a function not only of geometry but of displacement as well

$$[K_d] = [K_d(\vec{x}, \vec{u})]$$

Equation 3-61.

In the finite difference approximation, [K_d] should be approximated for the perturbed configuration as

$$\frac{\partial[K_d]}{\partial x_i} \cong \frac{[K_d(\vec{x}^0 + \Delta x_i, \vec{u}+\Delta\vec{u})] - [K_d(\vec{x}^0, \vec{u})]}{\Delta x_i}$$

Equation 3-62.

since the displacements are also implicit functions of the design variables. However, in NX Nastran, this displacement variation is ignored, resulting in

$$\frac{\partial [K_d]}{\partial x_i} \cong \frac{[K_d(\vec{x}^0 + \Delta x_i, \vec{u})] - [K_d(\vec{x}^0, \vec{u})]}{\Delta x_i}$$

Equation 3-63.

In most cases, Eq. 3-63 is a reasonable approximation. Eq. 3-62 is an often unnecessary, costly alternative. However, if the optimization results suggest that the sensitivities are not of sufficient accuracy, one (or both) of two options might be used. The first is to use the DOPTPRM entry to decrease DELB (the finite difference move parameter) by one or more orders of magnitude. This reduction might also be coupled with central difference approximations. (Setting the Bulk Data parameter CDIF to YES forces central differences.) The second option might be to reduce the move limits using DELP and DELX on the DOPTPRM entry and DELXV on the DESVAR entries. The goal of these changes is to improve the quality of the approximation seen by the optimizer.

Note

The parameter DSNOKD can be used to suppress the computation of the differential stiffness derivative. See Parameters for Design Sensitivity and Optimization.

Shape Sensitivity

Shape sensitivities are computed in Solution 200 whenever design variables are used to describe shape changes. They are computed automatically in connection with design optimization or if a sensitivity analysis alone (DSAPRT Case Control command or PARAM, OPTEXIT±4) is requested.

In shape optimization, the design responses are functions of the shape-defining design variables. Recall the reduced basis formulation used in connection with shape optimization

$$\{\Delta G\} = [T]\{\Delta x\}$$

Equation 3-64.

In Eq. 3-64, changes in grid coordinates $\{\Delta G\}$ are expressed in terms of changes in the design variables $\{\Delta x\}$, and the shape basis vectors $[T]$.

For semianalytic shape sensitivities, grid coordinates are perturbed a finite amount

$$\{\delta G\}_i = \frac{STPSCL}{E_i}\{T\}_i$$

Equation 3-65.

The subscript *i* in Eq. 3-65 characterizes the relation as a variation in shape for the *i*-th design variable.

E_i is the maximum strain energy norm computed using the $\{T\}_i$ basis vector. STPSCL is a scaling factor (default=1.0) that can be changed using the DOPTPRM Bulk Data entry.

The maximum strain energy norm used in Eq. 3-65 is computed by first determining the grid “forces” due to a shape basis vector “displacement”:

$$[K]\{T\}_i = \{F_s\}$$

Equation 3-66.

These forces can be used to compute a strain energy per grid as

$$\{F_s\}\{T\}_i = \{E\}_i$$

Equation 3-67.

Eq. 3-67 is an open product, resulting in a vector of strain energies at each grid. Locating the maximum term in the vector and taking its square root yields

$$E_i = (\max\{E\}_i)^{1/2}$$

Equation 3-68.

We get one such E_i for each design variable-shape basis vector pair. This has been found to yield a consistent set of finite difference move parameters for shape optimization.

Dynamic Response Sensitivities

Dynamic response sensitivities are computed in Solution 200 in connection with design sensitivity analysis and optimization. The following disciplines are available:

- Direct frequency
- Modal frequency
- Modal transient

See the DRESP1 Bulk Data entry description for a list of the response types supported for the above analysis disciplines.

General Considerations

The complete stiffness matrix for dynamic response analysis consists of a superposition from damping as well as direct matrix input:

$$[K] = [K_1] + ig[K_1] + i\sum g_e[K_e] + [K_2]$$

Equation 3-69.

where:

$[K_1]$	=	structural stiffness
g	=	uniform structural damping coefficient PARAM,G
g_e	=	structural damping coefficient on a MAT entry

$[K_2]$ = direct matrix input at grids DMIG

Currently a functional relation between design variables and *g* cannot be prescribed. The structural damping contributions are assumed constant for design sensitivity. They are accounted for in the analysis, but do not vary as a function of the design variables. Similarly, the direct matrix input quantities cannot be expressed as functions of the design variables since these are essentially external quantities. That is, they are determined based on considerations that are external to the design model. These quantities are also assumed to be constant.

The total damping force is due to a sum of the contributions from viscous elements $[B_1]$ and direct matrix input at the grids $[B_2]$

$$[B] = [B_1] + [B_2]$$

Equation 3-70.

For structural mass,

$$[M] = [M_1] + [M_2]$$

Equation 3-71.

The direct matrix input for damping and mass is assumed constant in the sensitivity analysis as is the case with the direct matrix input for stiffness.

Direct Frequency Response Sensitivities

Displacement sensitivities for direct frequency response are obtained by differentiating the governing equation below

$$(-\omega^2[M] + i\omega[B] + [K])\{u\} = \{P\}$$

Equation 3-72.

to obtain

$$(-\omega^2[M] + i\omega[B] + [K])\{\Delta u\} = \{\Delta P\} - (-\omega^2[\Delta M] + i\omega[\Delta B] + [\Delta K])\{u\}$$

Equation 3-73.

where Δ indicates differentiation with respect to a design variable, or $\partial/\partial x_i$. Eq. 3-73 is solved once for each forcing frequency, load condition, and design variable.

Limitations regarding {ΔP} Computation

Although the force derivative term (Δ*P*) is shown in Eq. 3-73 for completeness, this term is assumed to be zero in NX Nastran. This is often a good approximation. However, if gravity, thermal, or geometry-dependent loadings are significant, some error will be introduced into the approximation. This assumption also applies to the modal frequency and modal transient formulations. For all of the dynamic sensitivity solution sequences, the limitation of zero force sensitivity contribution should be kept in mind.

Modal Frequency Response Sensitivities

The governing equations for modal frequency response analysis are obtained from the equations for frequency response by again introducing the modal transformation of Eq. 3-79,

$$\{u\} = [\Phi]\{\xi\}$$

Equation 3-74.

to yield

$$(-\omega^2[\Phi]^T[M][\Phi] + i\omega[\Phi]^T[B][\Phi] + [\Phi]^T[K][\Phi])\{\xi\} = [\Phi]^T\{P\}$$

Equation 3-75.

Differentiating Eq. 3-75 with respect to a design variable and assuming that enough modes are retained such that

$$\{\Delta u\} \cong [\Phi]\{\Delta\xi\}$$

Equation 3-76.

is a good approximation yields, with the assumption of Eq. 3-82,

$$(-\omega^2[\Phi]^T[M][\Phi] + i\omega[\Phi]^T[B][\Phi] + [\Phi]^T[K][\Phi])\{\Delta\xi\} \cong$$
$$[\Phi]^T(\{\Delta P\} - (-\omega^2[\Delta M] + i\omega[\Delta B] + [\Delta K])\{u\})$$

Equation 3-77.

As with the case in transient analysis, Eq. 3-77 assumes that enough modes are retained so that the perturbed space is adequately spanned by the retained set of eigenvectors. That is, the error incurred by omitting the [Δφ]{ξ} term is assumed to be negligible.

With the derivatives of the modal coordinates computed from Eq. 3-77 , the displacement derivatives are available by direct substitution into Eq. 3-76.

Modal Transient Response Sensitivities

The governing differential equation for transient dynamic response is

$$[M]\{\ddot{u}\} + [B]\{\dot{u}\} + [K]\{u\} = \{P\}$$

Equation 3-78.

The modal transformation, used to reduce problem size, is given by

$$\{u\} = [\Phi]\{\xi\}$$

Equation 3-79.

where [φ] is a matrix whose columns are composed of a sufficient number of eigenvectors such that the reduced problem yields accurate results, and [ξ] is the vector of modal coordinates, or transformed displacements, which are solved for in the reduced problem, then yielding the actual displacements [*u*] from Eq. 3-79.

Introducing this modal transformation into Eq. 3-78 and differentiating with respect to a design variable yields the approximate (also see Eq. 3-82) expression for the modal transient displacement sensitivities

$$[\Phi]^T[M][\Phi]\{\Delta\ddot{\xi}\} + [\Phi]^T[B][\Phi]\{\Delta\dot{\xi}\} + [\Phi]^T[K][\Phi]\{\Delta\xi\}$$
$$\cong [\Phi]^T(\{\Delta P\} - ([\Delta M]\{\ddot{u}\} + [\Delta B]\{\dot{u}\} + [\Delta K]\{u\}))$$

Equation 3-80.

In Eq. 3-80 the assumption is made that a sufficient number of modes are included such that

$$\{\Delta u\} \cong [\Phi]\{\Delta\xi\}$$

Equation 3-81.

is a good approximation. This assumption implies that

$$[\Delta\Phi]\{\xi\} \cong 0$$

Equation 3-82.

Once the solution for the modal coordinates and their time derivatives is available, the displacement derivatives are computed from Eq. 3-81.

A Special Case of Frequency Response Sensitivities

Equivalent Radiated Power Response Sensitivities

Equivalent radiated power (ERP) is an output option used to quantify the noise level that radiates from a vibrating surface. ERP output is a measure of the normal frequency response velocity over the face of elements that comprise a panel by summing the elemental responses for all the elements that comprise the panel.

The equivalent radiated power attributable to the i^{th} element of a panel at frequency ω is given by:

$$ERPE_i(\omega) = \frac{1}{2} c \int_{S_i} v_n(\mathbf{x},\omega)\, v_n^*(\mathbf{x},\omega)\, ds$$

Equation 3-83.

where:

c	Scaling coefficient that is usually taken to be the product of density and speed of sound in the fluid medium.
$v_n(\mathbf{x},\omega)$	Normal velocity as a function of position at frequency ω.
$v_n^*(\mathbf{x},\omega)$	Complex conjugate of the normal velocity as a function of position at frequency ω.
S_i	Surface area of the i^{th} element.

Adjoint Load Method for ERP Sensitivity Analysis

When equivalent radiated power (ERP) is used as a design response, a single composite adjoint load for the entire panel is created so that the sensitivities (derivatives) are obtained directly.

The expression for the total derivative of the element ERP magnitude ($ERPE_i(\omega)$) with respect to the element nodal displacements gives the adjoint load for that particular element. Therefore, this derivative summed over the elements over which the ERP is found (as for a panel) gives the corresponding adjoint load for the ERP in question.

The adjoint load for the ERP magnitude on a single element is

$$\{p_{adj}\} = -4c\pi^2\omega^2 \sum_J h_J ([N]^T \{t\}\{t\}^T [N])\{u_r\}$$

Equation 3-84.

where:

J	The J^{th} Gauss point
h_J	The relevant weighted surface area for the J^{th} Gauss point
$[N]$	A matrix that converts the nodal displacements to the displacements at the Gauss point
$\{t\}$	A vector that transforms the Gauss point displacements to the surface normal displacement at the Gauss point
$\{u_r\}$	The real part of the element nodal displacements vector, $\{u_g\}$
c	A scaling coefficient, usually taken to be the product of density and speed of sound for the fluid medium
ω	The frequency, in cycles per unit time, at which the ERP magnitude is computed

The element ERP adjoint load becomes the basis for panel adjoint load for the ERP design response. The adjoint load for the panel is then used to find the ERP derivatives with respect to the design variables in the usual manner.

As for the panel ERP density derivatives, given that the ERP density for a panel is defined as

$$D_{panel} = (E / A)_{panel}$$

Equation 3-85.

where *E* is the ERP total magnitude over the panel, and *A* is the normal surface area, the derivative of *D* with respect to a single design variable is then

$$(dD / dx) = (1 / A)(dE / dx) - (1 / A^2)E(dA / dx)$$

Equation 3-86.

NX Nastran currently disregards changes in the panel normal surface area with changes in the design variables. Therefore, the second term drops out, and the derivative expression is simplified to

$$(dD / dx) = (1 / A)(dE / dx)$$

Equation 3-87.

where *dE*/*dx* is obtained with the adjoint load approach described above.

Aeroelastic Sensitivity Analysis

Aeroelastic response sensitivities are computed in Solution 200 in connection with design optimization. Sensitivities alone can be computed if the DSAPRT Case Control command is used or the Bulk Data parameter OPTEXIT is set to ±4. The following sensitivities are available:

- Static aeroelastic responses, including not only displacement, stresses, forces, etc., but also trim variables. This allows the evaluation of a structural change on the trim angle of attack.
- Stability derivative sensitivities. This, for example, allows the determination of the effect that strengthening a wing spar has on the rolling moment produced by a deflection of the aileron.
- Flutter sensitivities. These compute the changes in damping responses in a Flutter analysis at specified velocities due to changes in structural parameters.

This section contains a brief description of the theory for aeroelastic sensitivity analysis. For both theoretical and design modeling details as well as example problems, see the *NX Nastran Aeroelastic Analysis User's Guide*.

Static Aeroelasticity Sensitivity

The form of the equations are the same as for sensitivity analysis of static responses:

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

Equation 3-88.

where, for static aeroelastic analysis,

$$[K] = \begin{bmatrix} K_{ll} + K_{ll}^a & K_{lr} + K_{lr}^a & M_{ll}D + M_{lr} & K_{lx}^a \\ D^T M_{ll} + M_{rl} & D^T M_{lr} + M_{rr} & 0 & 0 \\ D^T K_{ll}^a + K_{rl}^a & D^T K_{lr}^a + K_{rr}^a & m_r & D^T K_{lx}^a + K_{rx}^a \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

$$[u] = \begin{Bmatrix} u_l \\ u_n \\ \ddot{u}_r \\ u_x \end{Bmatrix}$$

$$\{P\} = \begin{Bmatrix} P_l \\ 0 \\ P_r + D^T P_l \\ P_x \end{Bmatrix}$$

Equation 3-89.

This equation is differentiated with respect to a design variable and solved for the “displacement” sensitivities using a semianalytic approach as has been described in Eq. 3-46 for static sensitivity analysis.

Flutter Sensitivity Analysis

The eigenvalue problem for flutter is given by

$$\left[M_{hh}p^2 + \left(B_{hh} - \frac{1}{4}\rho\bar{c}V\frac{Q_{hh}^l}{k}\right)p + \left(K_{hh} - \frac{1}{2}\rho V^2 Q_{hh}^R\right)\right]\{u_h\} = \{0\}$$

Equation 3-90.

where the terms in the above equation are defined in the *NX Nastran Aeroelastic Analysis User’s Guide*.

Flutter sensitivity computes the rates of change of the transient decay rate coefficient γ with respect to changes in the design variables. γ is defined in connection with the complex eigenvalue p

$$p = \omega(\gamma + i) = p_R + p_I$$

Equation 3-91.

As with the other analysis disciplines, Eq. 3-90 is differentiated with respect to the design variables and solved for the quantity $\partial\gamma/\partial x_i$. The solution is semi-analytic in nature with derivatives approximated using either forward differences (which is the default if no shape-defining design variables are present) or central differences (PARAM,CDIF,YES, which is the default if shape variations are allowed).

Coupled Fluid-Structure Interaction Sensitivity

Coupled fluid-structure analysis is used to solve problems in which the interaction effects between fluid and structure are significant. NX Nastran uses a fully-coupled formulation for this analysis. Although its principal application is in acoustic problems, it can also be applied to other general problems, such as fluid storage (tank sloshing), or other situations in which an inviscid, irrotational formulation is valid. Refer to the *NX Nastran User's Guide* for details on the analysis formulation.

The sensitivity equations for fluid-structure interaction are identical in form to the dynamic response sensitivity equations and can be solved using the existing dynamic response sensitivity solution algorithms. To understand why this is so, the general equations of motion for the fluid-structure system can be written as

$$[M]\{\ddot{x}\} + [K]\{x\} = \{F\}$$

Equation 3-92.

The solution vector $\vec{x}$ consists of both structural displacements $\vec{u}$ and the fluid pressure $\vec{P}$.

$$\{x\} = \begin{Bmatrix} \{u\} \\ \{P\} \end{Bmatrix}$$

Equation 3-93.

The vector of external loads $\vec{F}$ is comprised of both structural as well as fluid forces or

$$\{F\} = \begin{Bmatrix} \{F_s\} \\ \{F_f\} \end{Bmatrix}$$

Equation 3-94.

The mass matrix in Eq. 3-92 is defined as

$$[M] = \begin{bmatrix} [M_s] & 0 \\ [A] & [M_f] \end{bmatrix}$$

Equation 3-95.

where:

$[M_s]$	=	structural mass
$[M_f]$	=	fluid mass (including compressibility effects)
$[A]$	=	fluid-structure coupling matrix

The stiffness matrix consists of

$$[K] = \begin{bmatrix} [K_s] & -[A]^T \\ 0 & [K_f] \end{bmatrix}$$

Equation 3-96.

where

$[K_s]$	=	structural stiffness
$[K_f]$	=	effective fluid stiffness

If the loads in Eq. 3-92 are frequency-dependent, a direct formulation can be written just as for the structural case where no fluid effects are present. Differentiating the direct frequency equation yields an expression similar to Eq. 3-73, which can be solved by exactly the same methods.

In very large problems, it is often desirable to apply the modal decomposition method in order to reduce the cost of the analysis. Modal formulations in coupled fluid-structure analysis are derived from a separate consideration of both fluid and structural components.

The structural modes are computed for a structure in a vacuum, that is, in the absence of any fluid effects. The modal transformation for the structural degrees of freedom can then be written as

$$\begin{aligned}[u] &= [\Phi_s]\{\xi_s\} \\ [m_s] &= [\Phi_s]^T[M_s]\,[\Phi_s] \\ [k_s] &= [\Phi_s]^T[K_s]\,[\Phi_s]\end{aligned}$$

Equation 3-97.

where

$[\Phi_f]$	=	the modes for structure in a vacuum

The modes for the fluid are computed under the effects of rigid wall boundary conditions, which eliminate the structural coupling effect. The resulting modal transformation for the fluid degrees of freedom is then

$$\begin{aligned}\{P\} &= [\Phi_f]\{\xi_f\}\\ [m_f] &= [\Phi_f]^T[M_f][\Phi_f]\\ [k_f] &= [\Phi_f]^T[K_f][\Phi_f]\end{aligned}$$

Equation 3-98.

where

$[\Phi_s]$	=	the modes for structure in a vacuum

After modal reduction for both structural and fluid degrees of freedom, the equations of motion can be written as

$$\begin{bmatrix} m_s & 0 \\ \left(\Phi_f^T A \Phi_s\right) & m_f \end{bmatrix}\begin{Bmatrix}\ddot{\xi}_s\\ \ddot{\xi}_f\end{Bmatrix} + \begin{bmatrix} k_s & -\left(\Phi_s^T A^T \Phi_f\right) \\ 0 & k_f \end{bmatrix}\begin{Bmatrix}\xi_s\\ \xi_f\end{Bmatrix} = \begin{Bmatrix}\Phi_s^T F_s\\ \Phi_f T F_f\end{Bmatrix}$$

Equation 3-99.

This modal transient formulation can be differentiated with respect to a design variable to obtain the equations for coupled fluid-structure interaction sensitivity. The form of the equations is similar to Eq. 3-80 for modal transient analysis, so it is not repeated here. However, the solution process is identical.

If the excitation is frequency-dependent rather than time-dependent, the resultant sensitivity equations assume the form given by Eq. 3-77 for structural problems. Due to the similarities of their equations, the solution process is again identical.

3.3 Optimization with Respect to Approximate Models

This section describes the process of optimization with respect to approximate models in NX Nastran. Many of these operations can be controlled with various parameter and Bulk Data selections.

To briefly review, once the finite element and sensitivity analyses are performed for the current design cycle, all the necessary information is available from which to construct the approximate model. This approximation is, in turn, used by the optimizer to search for a corresponding approximate optimum.

There are a number of parameters that enable you to exercise some control over the optimization phase of the design cycle. This section discusses these quantities with an emphasis on their relation to the overall process as well as guidelines for selecting default value overrides.

Approximate Optimization Control Parameters - An Overview

The modification of some of the quantities used to guide the approximate optimization may bring an increase in numerical efficiency, an enhanced usability of the final design, or a closer look at the optimization process by increasing the diagnostic printout. There are a number of such quantities that you may want to override. Some appear on the Bulk Data entries for design optimization, with the majority of the others available on the DOPTPRM Bulk Data entry.

Note

Since the DOPTPRM entry is optional, every value on it has a default. Thus, the use of this entry is always associated with some type of override.

The more common quantities, their basic function, and methods for accessing them are listed below:

APRCOD	Selects the type of approximation(s) to be used in the construction of the approximate model. The DOPTPRM entry is used to choose from one of the available methods by setting APRCOD to 1, 2, or 3 where 1 = direct linearization, 2 = mixed method (default), and 3 = convex linearization. See Approximation Concepts in Design Optimization for a theoretical overview.
CT, CTMIN	CT and CTMIN state the criteria that the optimizer uses to identify active and violated constraints. CT and CTMIN are specified on the DOPTPRM entry.
DELP, DPMIN	Move limits on properties for the approximate optimization. Since the approximations are only locally valid, these quantities, which are specified on the DOPTPRM entry, are used to limit the region of search during the approximate optimization. Move limits are discussed in this section.
DELX	Fractional change in any design variable during approximate optimization. This provides move limits on the design variables similar to the move limits on properties supplied by DELP and DPMIN. It can be changed using the DOPTPRM entry.
DELXV	Move limit for a particular design variable during approximate optimization. It is assigned on an individual design variable basis using the DESVAR Bulk Data entry. If not supplied, the default is provided using DELX on the DOPTPRM entry.
DESMAX	Maximum allowable number of approximate optimizations that may be performed. This is the same as the maximum allowable number of design cycles. Set on the DOPTPRM entry, this parameter provides an upper limit on design cycles, unless convergence is achieved before then.
DXMIN	Minimum design variable move limit. This absolute quantity ensures that reasonable move limits exist for numerically small design variables. This can be changed using the DOPTPRM entry.
IPRINT	Indicates the level of diagnostic printout from the approximate optimization phase. The default is 0 (no printout), and can range in value from 0 to 7. See Table 1 on the DOPTPRM entry.
METHOD	Indicates whether the modified method of feasible directions (METHOD=1), sequential linear programming (METHOD=2), or sequential quadratic programming (METHOD=3) is used. METHOD=1 is the default. METHOD is selected using the DOPTPRM entry. (See Glossary of Terms.)
PMAX	Maximum allowable property value specified on either a DVPREL1 or DVPREL2 entry. The set of all PMAX values acts as upper bounds on properties during the approximate optimization.

PMIN	Minimum allowable property value specified on either a DVPREL1 or DVPREL2 entry. The set of all PMIN values acts as lower bounds on properties during the approximate optimization.
XLB, XUB	Lower and upper bounds on a single design variable provided on the DESVAR entry. The optimizer will never propose a design that is outside of these bounds.

In addition to the above quantities, there are a number of internal optimizer parameters that may be changed as well. A complete listing of these parameters is in Table 1 of the DOPTPRM entry. Generally, there is little reason to modify them; however, the ability to do so may, at times, prove to be indispensable. Unless you are particularly well acquainted with numerical optimizers though, these parameters can usually be left unchanged and still yield quite satisfactory results.

Move Limits and Approximate Optimization

Since the approximations are only locally valid (see Approximation Concepts in Design Optimization and Eq. 1-6 in Structural Optimization), limits must be placed on the amount by which the design is allowed to vary during the approximate optimization. In NX Nastran, move limits are imposed in terms of both allowable changes in design variables and allowable changes in properties.

Imposing move limits using design variables is natural since the approximate model is explicit in these quantities. However, move limits on designed properties must also be considered as well, since the analysis model is a function of these properties. Moreover, these designed properties may be nonlinear functions of the design variables (DVPREL2 relations). A 10% change in a given design variable may equate to a 100% or more change in an analysis model property. Since these large design changes can easily invalidate the approximate model, move limits on properties must also be included.

During approximate optimization, each design variable is bounded from above and below by

$$x_i^L \le x_i \le x_i^U$$

Equation 3-100.

where x_i^L is the lower bound on the i-th design variable and x_i^U is the corresponding upper bound. These bounds are determined at the beginning of each design cycle according to

$$x_i^L = x_i^0 - \left|x_i^0\right| \cdot (\mathrm{DELX}, \mathrm{DELXV})$$
$$x_i^U = x_i^0 + \left|x_i^0\right| \cdot (\mathrm{DELX}, \mathrm{DELXV})$$

Equation 3-101.

where x_i^0 is the initial value of the i-th design variable. The allowable percentage change in the design variable is supplied by DELXV on the DESVAR Bulk Data entry. This value is optional and if not supplied, defaults to DELX, which is provided on the DOPTPRM Bulk Data entry. The DELX default is 1.0, or a 100% change in all design variables.

Note

If a DELXV is not present, the global parameter DELX provides move limits for all design variables. If a DELXV is provided, then it takes precedence by providing a specific move limit for a given design variable.

We have a similar situation for designed properties where each is bounded from above and below by

$$p_j^L \leq p_j \leq p_j^U$$

Equation 3-102.

where p_j^L is the lower bound on the *j*-th property and p_j^U is the corresponding upper bound.

These bounds are based on percentage changes in the property value p_j^0 at the outset of the approximate optimization. These bounds are computed as

$$\begin{aligned} p_j^L &= p_j^0 - \left|p_j^0\right| \cdot \mathrm{DELP} \\ p_j^U &= p_j^0 + \left|p_j^0\right| \cdot \mathrm{DELP} \end{aligned}$$

Equation 3-103.

where DELP is the maximum allowable percentage change in the property. DELP is defined on the DOPTPRM entry and has a default of 0.2 for a 20% maximum property variation.

Note

Move limits on properties are applied to every property referenced by the design model. There is no provision to make these limits property-dependent, that is, to apply a different DELP for different properties.

Eq. 3-101 and Eq. 3-103 effectively form a “box” around the current design. This effect is shown in Figure 3-8 where these move limits are shown for successive design cycles in a two design variable space. For the first cycle, the approximate optimum is found to lie at a corner of the box, where the objective is minimized and there are no active constraints. As a result of the second cycle, one of the constraints is slightly violated due to errors in the approximation. By the third cycle, a near optimal design has been found. Of course, the situation is usually more complex than in this simple design space. The intent of Figure 3-8 is merely to suggest some general features of the overall process.

Note

Even though the optimizer deals with an approximate model, note that the properties are always known precisely. That is, $p_j = p_j[x]$ is explicitly given by the design variable-to-property relations DVPREL1 and DVPREL2, and these relations are made available to the optimizer.

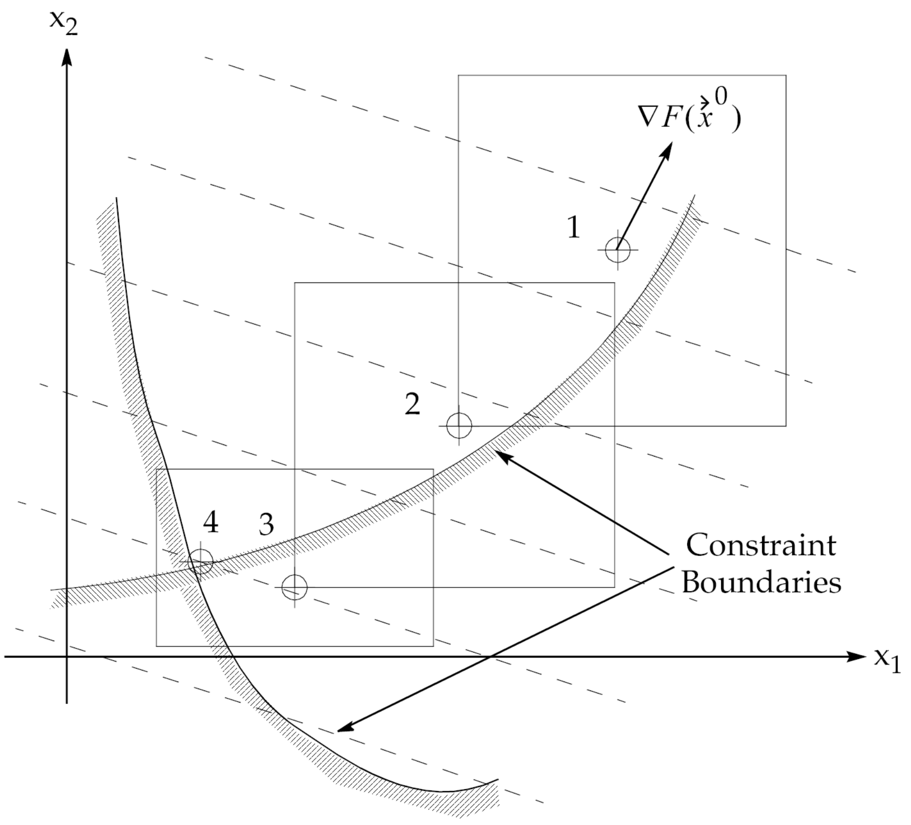


Figure 3-8. Sequence of Approximations

Avoiding Small Moves with DPMIN

The DELX default of 1.0 and the DELP default of 0.2 perform well in most cases, although these values may not always yield the greatest efficiency. If any of the design variables or properties are near zero, the move limits prescribed by Eq. 3-101 and Eq. 3-103 may tend to overly restrict design moves, and hence, the optimizer's progress. This can not only waste valuable computational resources, but the convergence decision logic may be "trapped" into believing that the design process has converged, when in reality the optimizer had just been unnecessarily restricted. To avoid this, minimum move limits are defined for both design variables and properties.

If any design variable is near zero such that $|x_i^0| \cdot (\text{DELX, DELXV}) << 1$, the minimum lower and upper limits are defined by

$$x_i^L = min[x_i^L(\text{DELX, DELXV}),\ x_i^0 - \text{DXMIN}]$$
$$x_i^U = max[x_i^U(\text{DELX, DELXV}),\ x_i^0 + \text{DXMIN}]$$

Equation 3-104.

where x_i^L (DELX, DELXV) and x_i^U (DELX, DELXV) are the lower and upper bounds from Eq. 3-101.

Note

The notation (DELX, DELXV) indicates that the default provided by DELX can be overridden if a DELXV is present. See Eq. 3-101.

Likewise, minimum move limits on properties are defined as

$$
\begin{aligned}
p_j^L &= min[\, p_j^L(\text{DELP}),\ (p_j^0 - \text{DPMIN})] \\
p_j^U &= max[\, p_j^U(DELP),\ (p_j^0 + DPMIN)]
\end{aligned}
$$

Equation 3-105.

where p_j^L(DELP) and p_j^U(DELP) are the lower and upper bounds from Eq. 3-103.

Recall that lower and upper limits exist for each design variable as defined by the XLB and XUB fields on the DESVAR Bulk Data entry. Regardless of the values computed in Eq. 3-101 and Eq. 3-104, move limits cannot be outside of these values. Considering all the effects, we have for the lower and upper bounds,

$$
x_i^L = max\left\{ min[\, x_i^L\ (\text{DELX, DELXV}),\ (x_i^0 - \text{DXMIN})],\ \text{XLB} \right\}
$$

$$
x_i^U = min\left\{ max[\, x_i^U\ (\text{DELX, DELXV}), (x_i^0 + \text{DXMIN})],\ \text{XUB} \right\}
$$

Equation 3-106.

For every designed property, we have a minimum lower bound PMIN and a maximum upper bound PMAX as defined on either a DVPREL1 or DVPREL2 entry. Considering these limits on properties leads to

$$
p_j^L = max\left\{ min[p_j^L(\text{DELP}), (p_j^0 - \text{DPMIN})], \text{PMIN} \right\}
$$

$$
p_j^U = min\left\{ max[p_j^U(\text{DELP}), (p_j^0 + \text{DPMIN})], \text{PMAX} \right\}
$$

Equation 3-107.

Figure 3-9 illustrates the situation for a property that is initially close to zero. $\bar{p}_j^L$ and $\bar{p}_j^U$ are the move limits resulting from Eq. 3-103. Since these limits are overly restrictive, the limits based on DPMIN, or p_j^L and p_j^U are used instead. Although not shown in the figure, if PMIN is a numerically greater quantity than p_j^L, it becomes the lower bound as indicated by Eq. 3-107. PMAX may also become the upper bound as well, if it is less than p_j^U.

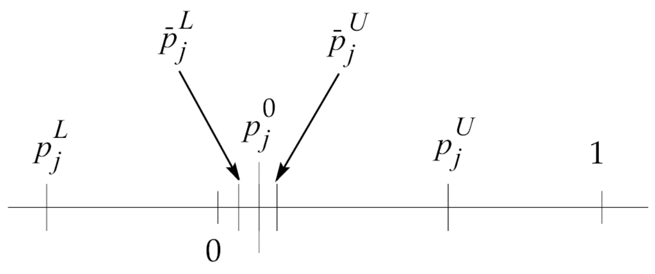


Figure 3-9. Move Limits on Properties

Automatic Updates of Move Limits

Parameters related to design variables can be changed using the DOPTPRM entry (DELX and DXMIN) and the DESVAR entry (DELXV, XLB, XUB). The designed property limits can be controlled using the DOPTPRM entry (DELP, DPMIN) and the DVPREL1 or DVPREL2 entries (PMIN, PMAX). This set of constants is used to recompute the move limits for each design cycle.

At times, it may be necessary for the code to automatically adjust these move limits if the problem becomes numerically ill-conditioned. The situation might arise as follows: An approximate problem is constructed, from which the optimizer determines a corresponding approximate optimum. Perhaps some of the approximate constraints are critical for this design. The responses are now evaluated by a finite element analysis, and it is determined that rather than just critical, these constraints are actually violated. Errors have thus been detected between the approximate and the true responses.

If this error continues from one design cycle to the next, it can be taken as an indication that the move limits are probably too wide. Continued constraint violations have an adverse effect on overall convergence. The move limit-controlling parameters are updated automatically in NX Nastran if the following criteria are satisfied:

- The current design cycle number is greater than or equal to three.
- There is at least one violated constraint (violated by more than 2%), and the level of constraint violation is increasing.

Under these conditions, DELP, DPMIN, DELX, and DXMIN are reduced by one-half of their current values. The reason for the first condition is that frequently the optimizer may violate the constraints somewhat as it makes favorable gains in the objective function in the first few cycles. However, if this condition continues, it may indicate that the problem is becoming ill-conditioned as a result of excessive move limits. A corresponding User Warning Message is printed as notification that this update has occurred (see Output Features and Interpretation). If the job is to be restarted, an updated DOPTPRM entry with the new move limits should be included in the restart deck.

Note

Under no conditions does an update increase the move limits. The only type of update is a reduction by one-half. The move limits cannot be increased again unless the engineer intervenes and manually changes them.

Implementation of Move Limits

Move limits on independent design variables are applied directly. Since the optimizer will never propose a value for a design variable outside of its bounds, the lower and upper move limits computed from Eq. 3-101, Eq. 3-104, and Eq. 3-106 are simply input directly to the optimizer.

Note

Move limits on dependent design variables are imposed as equivalent constraints, similar to that of properties in Eq. 3-108 below.

Move limits on properties, however, are implemented as equivalent constraints. For the *j*-th property in the design model, the equivalent lower and upper bound constraints are given by

$$g_L(\vec{x}) = \frac{p_j^L - p_j(\vec{x})}{\left|p_j^L\right|} \le 0$$

$$g_U(\vec{x}) = \frac{p_j(\vec{x}) - p_j^U}{\left|p_j^U\right|} \le 0$$

Equation 3-108.

where the lower and upper bounds p_j^L and p_j^U are given by Eq. 3-105 and Eq. 3-107.

Eq. 3-108 is necessary because the optimizer modifies design variables and not properties. Of course, the property values are known from the DVPREL1 and DVPREL2 relations. For any given set of design variables, the optimizer can use Eq. 3-108 to determine whether or not the corresponding properties are within their allowable limits.

Relaxation of Move Limits

Since a small numerical constraint violation is allowed by the optimizer (by default), move limits on properties may not be satisfied with equality at their bounds. This is usually of little concern except when the design is at PMIN or PMAX in Eq. 3-107. Occasionally, these bounds might be slightly violated in a numerical sense but usually this is of little consequence.

A puzzling situation may occur if the current design is infeasible. If this is the case, the optimizer's primary task is to reduce the level of constraint violation to find a feasible solution if one exists. Since the move limits on properties are implemented as equivalent constraints, the optimizer does not know the difference between response-type and move limit-type constraints. Thus, by seeking to reduce the level of constraint violation, some of these move limits may be violated. For initial designs with highly violated constraints, it is not unusual to see analysis model properties vary by 25% or 30% even though the move limits may actually only be 20%. This usually does not pose any practical difficulties, but if it does, the initial move limits can always be reduced accordingly using the DOPTPRM entry. (In rare instances this optimization feature may yield a physically meaningless property value (e.g., a negative plate thickness). In this case the analysis cannot proceed and a fatal message will be issued. The remedy, should this occur, is to restart with a reduced value of DELP on the DOPTPRM entry.)

Using Design Variable Bounds to Enforce Move Limits on Properties.

In contrast, the bounds on design variables are always honored by the optimizer. Since the design variables are the quantities that the optimizer varies directly (and thus has direct control over), their bounds are never exceeded. To implement bounds on properties that are critical from a design standpoint, it is recommended that a linear design variable to property relation (DVPREL1) be prescribed if possible with appropriate bounds on the design variable (stated on the DESVAR entry). Since the design variable bounds are always enforced, the corresponding property then never exceeds its lower and upper bounds. (An exception here is with respect to dependent design variable bounds, which are always treated as constraints.)

Numerically Identifying the Active and Violated Constraints

A numerical optimizer must use numerical criteria to judge when a constraint is active and when it is violated. These criteria are established using the values of CT and CTMIN, which appear on the DOPTPRM entry.

These quantities are shown in Figure 3-10 for a single inequality constraint in a simple two-design variable space. A constraint is considered active if its numerical value exceeds CT. Once a constraint is active, its gradient is included in the search direction computation. See Appendix D on numerical optimization for further details. An active constraint may subsequently become inactive if its value falls below CT.

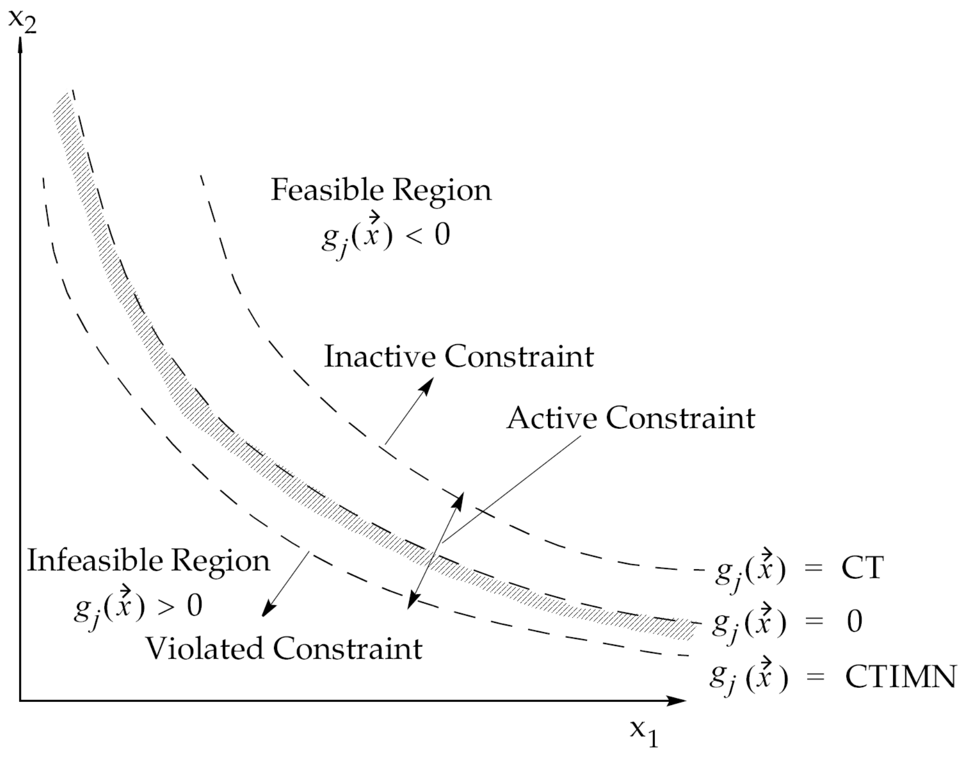


Figure 3-10. Active and Violated Constraints

The optimizer identifies a constraint as violated if its value is greater than CTMIN. The default for CTMIN is 0.003 as is seen in Figure 3-11. Thus, some small constraint violation (three-tenths of one percent, by default) is tolerated. Note that this establishes a numerical constraint boundary that is not identical to the exact constraint boundary.

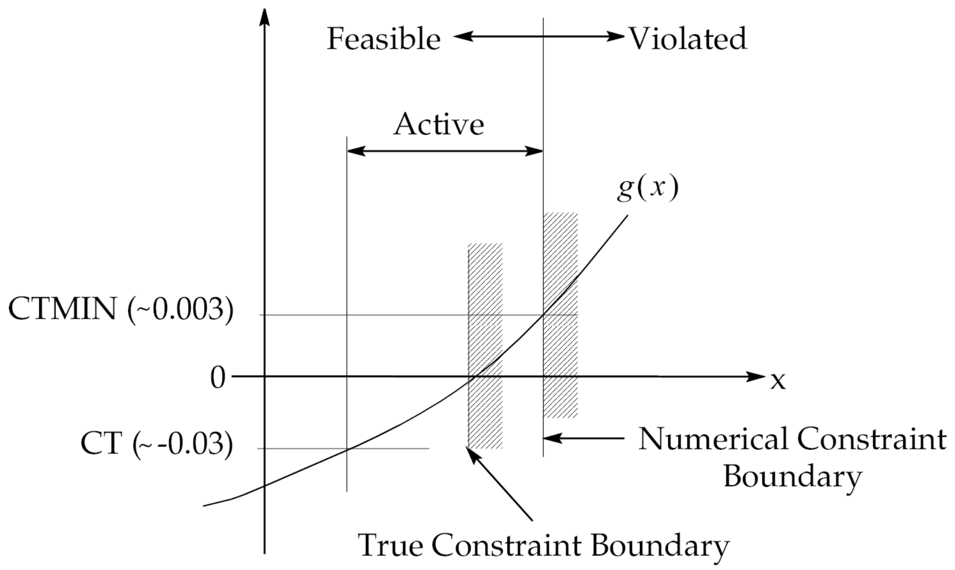


Figure 3-11. CT and CTMIN

Optimizer Convergence Parameters

In this section, we describe parameters that you can modify to affect convergence (although experience indicates that there is seldom a need to vary these parameters.) All of these may be changed using the DOPTPRM entry; see the *NX Nastran Quick Reference Guide* for more information.

Parameters related to convergence at the optimizer level include the following:

DABOBJ	Absolute change in the objective to denote convergence at the optimizer level.
DELOBJ	Relative change in the objective function to denote convergence at the optimizer level. The combination of DABOBJ and DELOBJ is used to identify the diminishing returns associated with convergence at the optimizer level.
ITMAX	Maximum number of iterations allowed within the optimizer (default = 40). This value is provided so that poorly converging problems are caught.
ITRMOP	The number of consecutive iterations for which the DABOBJ, DELOBJ convergence criteria must be satisfied before convergence is indicated.
ITRMST	The number of consecutive iterations for which the convergence criteria must be satisfied to indicate convergence in the sequential linear programming method. This is equivalent to ITRMOP and defines the maximum number of approximate linear problems that may be solved.
JTMAX	Maximum number of iterations allowed in the sequential linear programming method, used only if METHOD = 2 (see DOPTPRM entry). Its function is equivalent to ITMAX, used for the modified method of feasible directions.

3.4 Convergence Tests

Since structural optimization is an iterative process, numerical criteria must be established to determine when the overall process has converged. There are two levels at which convergence is tested: the first and lower level is at the optimizer level; the second and higher level is with respect to the overall design cycles. This section is concerned with design-cycle level convergence. Convergence at the optimizer level is briefly discussed at the end of the previous section and in greater detail in Introduction.

The numerical values of all of the convergence criteria discussed in this section can be changed using the DOPTPRM Bulk Data entry. This entry is optional, though, since defaults for all of these quantities have been provided. However, it is recommended that you always check to ensure their validity for the problem at hand. The trade-off between analysis cost and acceptable design accuracy should always be of prime consideration.

Convergence of Design Cycles: Hard and Soft Convergence

Two methods are used to test for convergence with respect to overall design cycles. These methods are denoted as soft convergence and hard convergence. Soft convergence is based on the results of the approximate optimization, while hard convergence is based on finite element analysis results.

Recall that at the outset of each design cycle, a finite element analysis and a sensitivity analysis are performed. The approximate model is then constructed followed by an optimization with respect to this approximation. Once a corresponding approximate optimum is found, the proposed design is submitted for another finite element analysis to compute the updated responses, thus marking the beginning of the next design cycle.

Hard convergence compares the results of this most recent finite element analysis with those from the previous design cycle. Since this test compares the exact results (within the limits of the finite element approximations) from two consecutive analyses, the conclusions are said to be based on hard evidence. Since this test is conclusive, this is the default test for determining whether or not to terminate the design-cycle process.

Soft convergence compares the design responses from the approximate optimization with those obtained from the previous finite element analysis. This test, although not as conclusive as hard convergence, is often an acceptable criterion to indicate convergence. For example, if a particular analysis model has a high solution cost and the optimizer has not changed the design to any large degree, it may be permissible to forego the final finite element analysis. The results from the previous analysis are probably of sufficient validity for design purposes; therefore, soft convergence may be a preferable and cost effective exit point.

Note

Soft convergence does not terminate the design cycle process unless the parameter SOFTEXIT is explicitly set to ‘YES’. The default for SOFTEXIT is ‘NO’.

These convergence tests are shown in the simplified flowchart of Figure 3-12. On the initial cycle, the hard convergence test is skipped and program flow proceeds directly to the sensitivity analysis and approximate optimization. Once an approximate optimum is obtained, a soft convergence test is performed to compare these results with the prior finite element analysis results. However, the design process is stopped only if soft convergence is indicated and the parameter SOFTEXIT has been set to ‘YES’. This test is performed regardless of whether the parameter SOFTEXIT is set to

'YES'. The printout for each design cycle includes a status report on the soft convergence test; see Output Features and Interpretation for examples of this output.

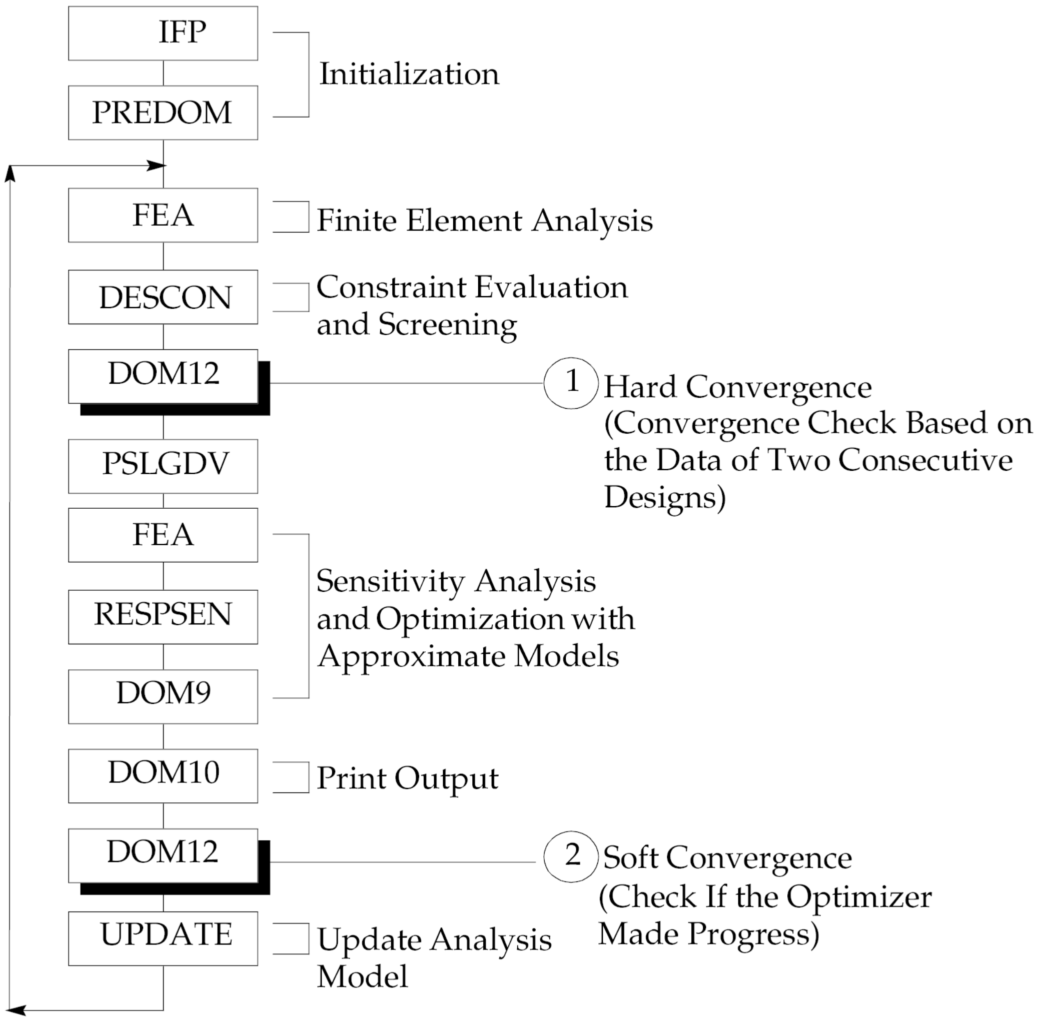

IFP
PREDOM
Initialization
FEA
Finite Element Analysis
DESCON
Constraint Evaluation and Screening
DOM12
1
Hard Convergence (Convergence Check Based on the Data of Two Consecutive Designs)
PSLGDV
FEA
RESPSEN
DOM9
Sensitivity Analysis and Optimization with Approximate Models
DOM10
Print Output
DOM12
2
Soft Convergence (Check If the Optimizer Made Progress)
UPDATE
Update Analysis Model

Figure 3-12. Convergence Testing and Program Flow

If soft convergence has not terminated the design process, a finite element evaluation of the new proposed design is performed followed by constraint evaluation and screening. If the analysis results are not appreciably different from those of the prior cycle, hard convergence is achieved, and the design cycle process is terminated. If the hard convergence criteria are not satisfied, the design cycle process continues.

Maximum Number of Iterations (Design Cycles) DESMAX

If neither hard nor soft convergence is achieved, the design process will continue until convergence is indicated or until the maximum allowable number of iterations (design cycles) is met. This default maximum number of iterations is five but can be changed on the DOPTPRM entry using DESMAX.

The following paragraphs flowchart and discuss the soft and hard convergence decision logic.

Soft Convergence Decision Logic

A schematic of the soft convergence decision logic is shown in Figure 3-13. The result of the test is a true or false value for the logical variable SOFTCV. The design cycles terminate only if SOFTCV is ‘TRUE’ and the parameter SOFTEXIT is ‘YES’. No other combinations allow soft convergence to halt the design-cycle process.

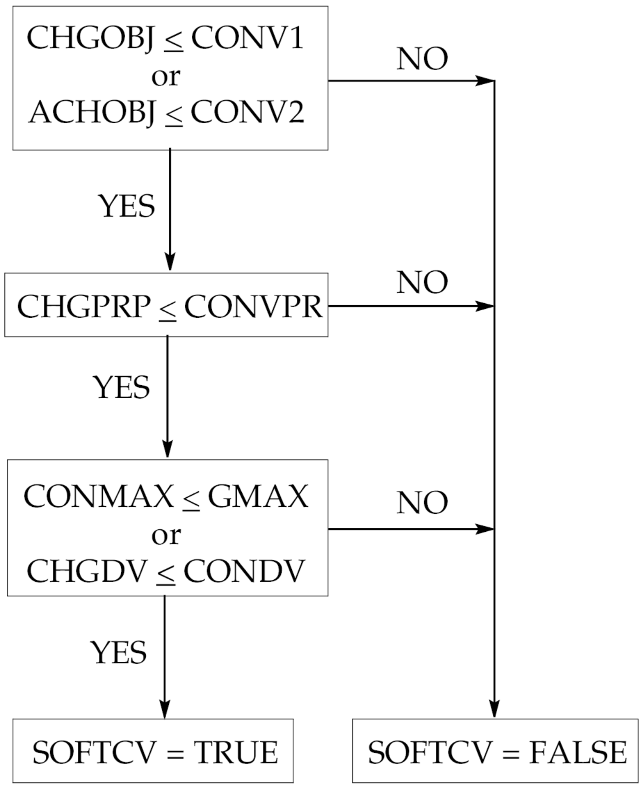


Figure 3-13. Soft Convergence Decision Logic

From Figure 3-13 , the first test is a check of the relative and absolute changes in the objective. The definitions of these parameters appear in Table 3-1 along with their default values. These values may be modified using the DOPTPRM entry and also apply to the hard convergence checks.

Table 3-1. Convergence Criteria Parameters

Internal Variable	Definition	Parameters	Default
CHGOBJ	$\left\lvert \dfrac{OBJ^{(P)} - OBJ^{(P-1)}}{OBJ^{(P-1)}} \right\rvert$	CONV1	0.001
ACHOBJ	$\left\lvert OBJ^{(P)} - OBJ^{(P-1)} \right\rvert$	CONV2	0.01

Table 3-1. Convergence Criteria Parameters

Internal Variable	Definition	Parameters	Default
CHGPRP	$\underset{1 \le i \le NPROP}{max} \left\lvert \frac{P_i^{(P)} - P_i^{(P-1)}}{P_i^{(P-1)}} \right\rvert$	CONVPR	0.001
CHGDV	$\underset{1 \le i \le NDV}{max} \left\lvert \frac{x_i^{(P)} - x_i^{(P-1)}}{x_i^{(P-1)}} \right\rvert$	CONVDV	0.001
CONMAX	$\underset{k}{max} \{g_k(\vec{x})\}$	GMAX	0.005

If either of the relative or absolute changes in the objective is satisfied, the next check is to see whether or not the analysis model properties have changed appreciably. If the changes in the objective and analysis model properties are negligible, the subsequent check determines if the design constraints are satisfied. If these constraints are satisfied, then soft convergence to a feasible design is achieved.

If some of the constraints are still not satisfied, a check on the change in design variables is performed. The conclusion here is that if the design variables, properties, and objective function are unchanged by the optimizer, then convergence is indicated although to an infeasible design. Since the optimizer cannot make any further progress, an additional finite element analysis is probably unnecessary.

Hard Convergence Decision Logic

Hard convergence is always used to decide whether or not the design cycle process should continue, provided that the maximum allowable number of design cycles (DESMAX) is not yet reached. The satisfaction of hard convergence is always sufficient to stop the design process.

A flowchart of these criteria is shown in Figure 3-14 with the corresponding definitions of the terms appearing previously in Table 3-1. Even though this test may indicate convergence, note that one of three possible conclusions may be reached.

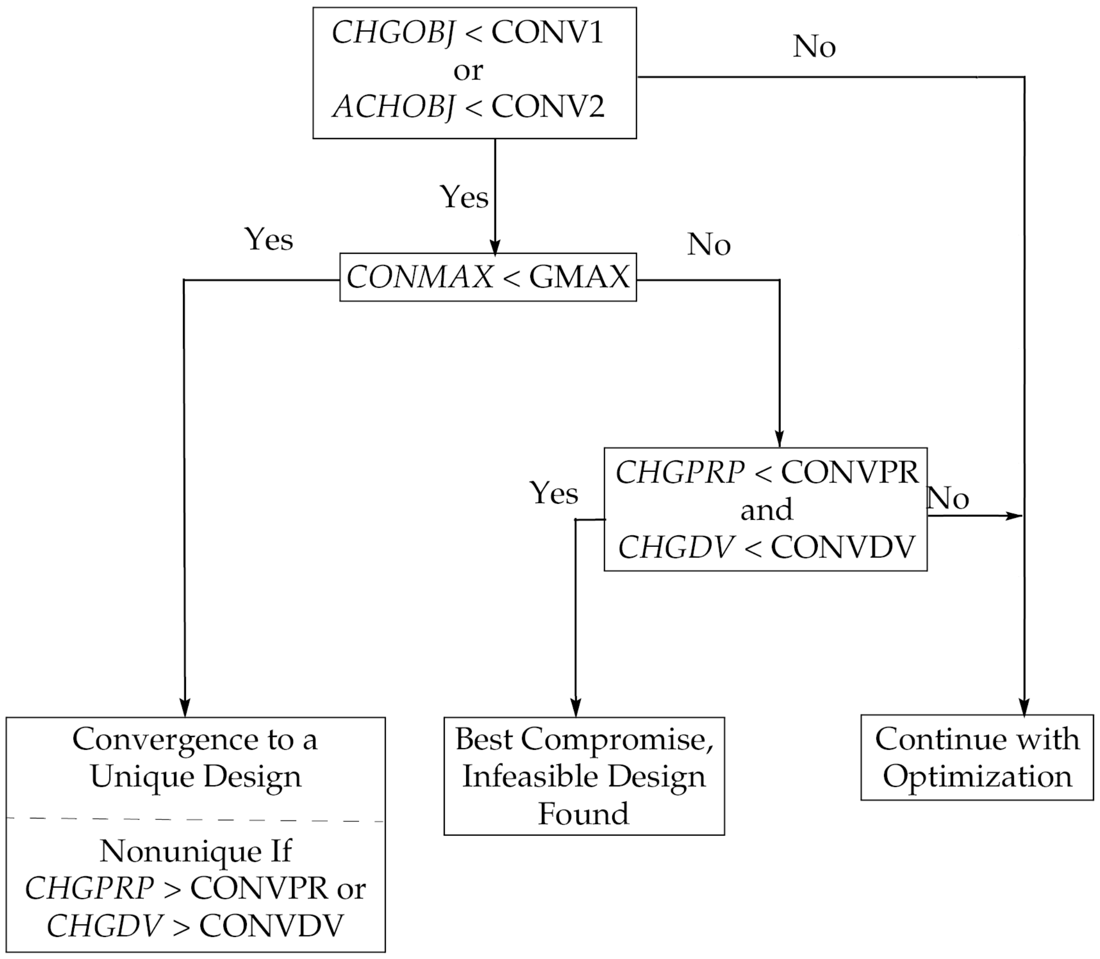

CHGOBJ < CONV1
or
ACHOBJ < CONV2
No
Yes
Yes
CONMAX < GMAX
No
Yes
CHGPRP < CONVPR
and
CHGDV < CONVDV
No
Convergence to a Unique Design
Nonunique If *CHGPRP* > CONVPR or *CHGDV* > CONVDV
Best Compromise, Infeasible Design Found
Continue with Optimization

Figure 3-14. Hard Convergence Decision Logic

As with soft convergence, the first check is with respect to the relative and absolute changes in the objective function. The reasoning behind the application of an 'OR' test is based on a consideration of the objective function magnitude. If the objective is large, such as the weight of a heavy machine part in kilograms, converging to within a plus or minus range of ten kilograms is probably quite sufficient. Forcing convergence to within a fraction of a kilogram may be meaningless as well as expensive. On the other hand, there are situations where the objective function is a very small number (e.g., minimization of the difference between analysis and test results). For such cases, minimization of the absolute change may be more meaningful than the relative change.

If the objective has not changed appreciably in either an absolute or a relative sense, a check is performed to ensure that the maximum constraint value is less than its maximum allowable. If this criterion is satisfied, then hard convergence is achieved. However, you should check to see if the relative changes in properties CHGPRP or the relative changes in design variables CHGDV are satisfied. If they are satisfied, then the design process has converged to a unique design. If not, a nonunique design may have been found.

Convergence to a Nonunique Optimum

From a design perspective, convergence to a nonunique design provides useful information. In many cases, this can indicate that the properties found at the end of the optimization process may be manually further modified by the engineer with correspondingly little change in the objective function and/or constraints. (In other words, the “optimal” design can be expected to exhibit low sensitivity with respect to changes in the design variables.) A design space of this type is shown in Figure 3-15 where contours of the objective and active constraint boundary have similar curvatures in the region of the optimum. In situations such as these, the design may be modified to take advantage of available sheet metal gauges or tube sizes without violating the active constraint(s) and with correspondingly little change to the optimum objective. Of course, any proposed changes should be subjected to an analysis and the results checked carefully to ensure that other performance constraints are not violated.

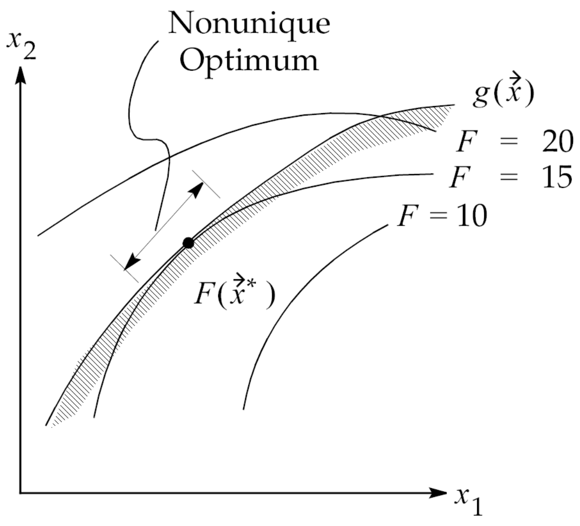


Figure 3-15. Nonunique Optimum

Returning to the hard convergence decision logic of Figure 3-14 , note that the code only checks the relative changes in properties and design variables if the limit on the maximum constraint value is not satisfied. The purpose of this check is to determine whether the design is changing for the case of violated constraints. If the design is still varying, we are justified in continuing optimization in order to try to overcome the constraint violation(s). However, if the design is not changing, then we are at a point in the design space that represents a best compromise solution among the violated constraints. This situation is shown in Figure 3-16 where the optimal design is the one that minimizes the sum of the constraint violation.

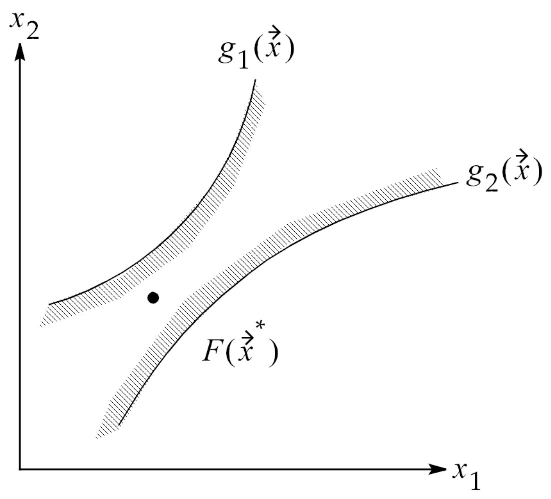


Figure 3-16. Design Space, No Feasible Solutions

Given the implications of hard convergence, a careful review of the output is advised in order to determine which convergence conditions have been met. Just because the process has been terminated does not necessarily imply that a unique, feasible design has been found.

Caution

Always check and confirm the conditions under which convergence is achieved.

Chapter 4: Input Data

This chapter describes all Executive Control, Case Control, Bulk Data, and parameter requirements for design sensitivity and optimization. These various input formats enable the following:

1. Specification of the applicable analysis discipline(s).

2. Definition of the design model.

3. Overrides of the optimizer's internal parameters.

4. Control of the program flow and results output.

Use of the material in this chapter is covered in Design Modeling for Sensitivity and Optimization and Example Problems.

4.1 File Management Section, Executive Control

Solution 200

In design sensitivity and optimization, the only required Executive Control statement is the SOL statement:

```
SOL 200
```

This states that subDMAP DESOPT is to be invoked, which is the main subDMAP for design sensitivity and optimization. The following section lists the supported analysis types in Solution 200; these are chosen using the Case Control command, ANALYSIS.

Note

Solution Sequences 108, 111, and 112 had previously supported dynamic response sensitivities. This capability has been moved from these solution sequences to Solution 200.

Shape Optimization and the File Management Section

File Management Section statements (see File Management Statements in the *NX Nastran Quick Reference Guide* for more information) are required in SOL 200 for:

- Shape optimization for the direct input of shapes method.

  This approach DBLOCATEs a set of displacement vector data from a database and uses this data to generate a set of shape basis vectors for design optimization. The following statements must be used:

```
ssign f1 = 'file.MASTER'
dblocate datablk=(ug/ugd,geom1/geom1d,geom2/geom2d), logical=f1
```

- Connecting with external programs to compute DRESP3 type responses.

  With this approach, you use the CONNECT statement to connect a user-defined external program that evaluates DRESP3 responses.

The filename file.MASTER is the name of the database for the auxiliary model analysis. The filename "file" is arbitrary. The UG, GEOM1, and GEOM2 data blocks must be DBLOCATEd and renamed to UGD, GEOM1D, and GEOM2D, respectively. These new names cannot be changed since the code looks for the data in these locations explicitly. See Relating Design Variables to Shape Changes and Shape Optimization of a Culvert for a discussion of, and examples using, this method.

4.2 Case Control Section

In NX Nastran analysis, the Case Control Section is used to specify the applied loads, boundary conditions, and subcases, and to request the form and type of analysis output. The additions to the Case Control for design optimization are few and do not require any modification of the Case Control commands already required for analysis.

Case Control commands for design sensitivity and optimization are associated with the following four tasks:

- Analysis discipline definition.
- Design model definition.
- Design response characterization.
- Shape basis vector computation.

This section discusses each of these tasks individually.

Analysis Discipline Definition

Design optimization in NX Nastran is multidisciplinary: a number of different analyses may be performed in Solution 200, and the results used simultaneously in optimization.

The analysis types are defined on a subcase basis using the ANALYSIS Case Control command. It can be set to any of the following values:

$$\text{ANALYSIS} = \left\{ \begin{array}{l} \text{STATICS} \\ \text{MODES} \\ \text{BUCK} \\ \text{DFREQ} \\ \text{MFREQ} \\ \text{MTRANS} \\ \text{SAERO} \\ \text{FLUTTER} \end{array} \right.$$

The following are a few Case Control issues to be aware of when using the ANALYSIS command.

- If ANALYSIS is specified above the subcase level, all subsequent subcases apply to that analysis type until redefined in a later subcase.
- The buckling subcase defined by ANALYSIS = BUCK refers to the first STATICS subcase.
- Multiple boundary conditions may be used in Solution 200 for ANALYSIS = STATICS.

- Limited multiple boundary condition support exists for ANALYSIS = MODES. A normal modes subcase can have its own unique set of boundary conditions, but only one normal modes subcase can exist in a Solution 200 run.

Design Model Definition

The design model definition process in Case Control includes identification of the design objective function and the design constraint sets.

Design Objective Identification

The design objective is identified in Case Control with the command:

$$\mathrm{DESOBJ}\begin{pmatrix}\mathrm{min}\\\mathrm{max}\end{pmatrix} = \mathrm{n}$$

Equation 4-1.

where n is the set identification of a design response on either a DRESP1 or DRESP2 Bulk Data entry. This response must be a single, scalar quantity.

A DESOBJ entry appearing above the subcase level identifies a “global” response. Weight and volume are typical examples of global responses. DESOBJ can also be used at the subcase level if the design goal is to minimize or maximize a subcase-dependent response.

Design Constraint Identification

Design constraint sets are identified in Case Control by the following commands:

```
DESGLB = n
DESSUB = n
```

where n is the set identification number of a DCONSTR or a DCONADD Bulk Data entry.

DESGLB is used above the subcase level to define a subcase-independent constraint set. Weight and volume are subcase-independent responses, as are DRESP2 responses that are not functions of DRESP1 responses (e.g., DRESP2s that are functions of design variables, table constants, and grid coordinates only).

DESSUB defines subcase-dependent constraint sets at the subcase level. A DESSUB command remains in effect until replaced with a new DESSUB in a subsequent subcase.

The overall design constraint definition process is:

1. Identify design responses in the Bulk Data using DRESP1 and/or DRESP2 entries.

2. Impose limits on these responses using DCONSTR Bulk Data entries.

3. Combine these DCONSTR sets, if desired, using DCONADD Bulk Data entries.

4. Identify the DCONSTR/DCONADD set identification numbers in Case Control using DESGLB for globally-applied constraints and DESSUB for subcase-dependent constraints.

Design Response Characterization

Strictly speaking, Case Control output requests are unnecessary in design sensitivity and optimization. In a number of instances they can, however, be quite useful.

Data Recovery in Solution 200

Solution 200 builds its own internal Case Control for data recovery based on the list of DRESP1 responses identified in the design model. This provision ensures that all necessary data recovery is always performed for design sensitivity and optimization. However, if you want to view NX Nastran results, Case Control output requests are necessary. The frequency of this output with respect to design cycle number is controlled by the Bulk Data parameter NASPRT.

Case Control and Design Responses

Case Control output commands can often be used to resolve design response ambiguities. For example, element stresses can be output using either von Mises or maximum shear. (The ambiguity arises since both share the same plot code ID referenced on the DRESP1 entry. See the *NX Nastran Quick Reference Guide* for a list of these plot codes.) The form identified in Case Control is taken as the representation for design sensitivity and optimization. For example,

```
STRESS( VONMISES )    = 15   (von Mises stresses used for analysis and optimization)
STRESS( SHEAR )       = 15   (maximum shear stresses used for analysis and optimization)
```

If left unspecified, the defaults will be used (which in this case, is the von Mises representation). See the *NX Nastran Quick Reference Guide* for defaults.

Dynamic responses can be computed using either a real/imaginary or a magnitude/phase form. Here again, Case Control can be used to define the output representation. For example,

```
STRAIN( IMAG )     = ALL    (real/imaginary form for analysis and optimization)
STRAIN( PHASE )    = ALL    (magnitude/phase form for analysis and optimization)
```

Output Frequencies and Output Time Steps

In design optimization, design responses in dynamic analysis are computed for all output frequencies (ANALYSIS = DFREQ or MFREQ) and time steps (ANALYSIS = MTRAN). This can lead to a huge amount of data. Case Control can be used to limit this output and save computational resources by defining output sets with

```
    OFREQ = n
    OTIME = m
```

where n and m are frequency and time sets, respectively.

Design responses will only be computed for the frequencies and/or time steps identified in the above sets.

Shape Basis Vector Computation

Additions to Case Control for shape optimization are only necessary when using the analytic boundary shapes method.

Recall from Relating Design Variables to Shape Changes, that the analytic boundary shapes approach uses additional Bulk Data Sections to describe the auxiliary boundary models. Each of these additional sections must have a corresponding Case Control.

Each of the auxiliary boundary model Case Control Sections are identified by the delimiter

```
AUXCASE
```

The auxiliary boundary model Case Control Sections must follow the Case Control Section for the primary model.

Within the auxiliary boundary model Case Control Section, individual auxiliary models are identified using

```
AUXMODEL = n
```

where n is the auxiliary boundary model ID, referenced in the corresponding BEGIN BULK = n Bulk Data Section delimiter. An example using AUXCASE and AUXMODEL is shown on 70 in Relating Design Variables to Shape Changes.

4.3 Bulk Data Entries

The design model is defined in the Bulk Data Section. This definition includes the design variables, the design variable-to-property relations, shape basis vectors, design constraints, and so on. In short, all of the quantities related to the basic optimization problem statement, Eq. 1-1 through Eq. 1-5 in Getting Started are defined here.

This section contains a brief listing and description of all the Bulk Data entries related to design sensitivity and optimization. This information has been extracted from the complete Bulk Data listings in Bulk Data Entries . For each entry in this section, related entries have also been listed with appropriately shaded fields to help clarify the interrelations among the various data.

DCONADD

Purpose:

Defines a new constraint set as a union of DCONSTR entry sets.

Entry Description:

DCONADD	DCID	DC1	DC2	DC3	etc.		

Field	Contents
DCID	Design constraint set identification number. (Integer > 0).
DCi	DCONSTR entry identification number. (Integer > 0).

Associated Entries:

DCONSTR	DCID	RID	LALLOW	UALLOW				

Discussion:

Constraints must be selected in Case Control. The DCONADD entry allows a number of DCONSTR entry sets to be unified into a single set, making their selection in Case Control easier.

DCONSTR

Purpose:

Places limits on a design response. When selected in Case Control by either DESGLB or DESSUB, the DCONSTR sets define the design constraints.

Entry Description:

DCONSTR	DCID	RID	LALLOW	UALLOW				

Field	Contents
DCID	Design constraint set identification number (Integer > 0).
RID	DRESPi entry identification number (Integer > 0).
LALLOW	Lower bound on the response quantity (Real, Default = -1.0E20).
UALLOW	Upper bound on the response quantity (Real, Default = 1.0E20).

Associated Entries:

The DRESP1 and DRESP2 entries identify responses that may be constrained.

DRESP1	ID	LABEL	RTYPE	PTYPE	REGION	ATTA	ATTB	ATT1
	ATT2	-etc.-						

DRESP2	ID	LABEL	EQID	REGION				
	“DESVAR”	DVID1	DVID2	-etc.-				
	“DTABLE”	LABL1	LABL2	-etc.-				
	“DRESP1"	NR1	NR2	-etc.-				
	“DNODE”	G1	C1	G2	C2	-etc.-		

The DCONADD entry can be used to form the union of a number of DCONSTR sets.

DCONADD	DCID	DC1	DC2	DC3	-etc.-			

Discussion:

During the optimization process, material may be removed from and/or added to the structure to achieve the optimum objective. As this occurs, the response of the structure to the applied loads changes. Member stresses, joint displacements, natural frequencies, etc., are altered by the changes that the optimizer is making.

It is important, then, to place bounds on the structural responses so that the optimized structure responds in a manner that is acceptable to the designer. For example, it is common to place upper and lower bounds on the member stresses so that the maximum allowable stresses are not exceeded. Joint displacement, buckling load factors, and natural frequencies might also be bounded.

When a bound is placed on a structural response, this condition is called a constraint. Both first and second-level responses may be constrained.

To constrain a given response quantity, a DCONSTR (Design CONSTRaint) entry is used. This entry, in turn, points to a specific response ID identified on either a DRESP1 or a DRESP2 Bulk Data entry. Note that the DCONSTR entry serves only to place bounds on a response defined elsewhere in the design model data. These bounds are used to construct normalized constraints for use by the optimizer as discussed in Defining the Constraints .

DCONSTR entry sets (or the DCONADD union of these sets) must be selected in the Case Control Section to complete the constraint definition process. The Case Control command DESGLB is used to identify subcase-independent constraints (global constraints), while the DESSUB Case Control command is used to select subcase-dependent constraint sets.

DEQATN

Purpose:

Defines equations for use in synthetic relations. These equations can be used to define either second-level responses or second-level design variable-to-property relations.

Entry Description:

DEQATN	EQID	EQUATION
	EQUATION (Cont.)	

Field	Contents
EQID	Unique equation identification number. (Integer > 0).
EQUATION	Equation(s) (Character).

Associated Entries:

The equation ID may be referenced in connection with a second-level response on a DRESP2 entry or in the definition of a synthetic property relation on a DVPREL2 entry.

DRESP2	ID	LABEL	EQID	REGION				
	“DESVAR”	DVID1	DVID2	-etc.-				
	“DTABLE”	LABL1	LABL2	-etc.-				
	“DRESP1"	NR1	NR2	-etc.-				
	“DNODE”	G1	C1	G2	C2	-etc.-		

DVPREL2	ID	TYPE	PID	FID	PMIN	PMAX	EQID	
	“DESVAR”	DVID1	DVID2	DVID3	-etc.-			
	“DTABLE”	LABL1	LABL2	LABL3	-etc.-			

Discussion:

A unique feature of NX Nastran design sensitivity and optimization is that it allows the engineer to create new response quantities and nonlinear design variable-to-property relations. This is accomplished by defining equations much like the function definition procedure of many programming languages.

The DEQATN entry is used to specify a single equation or a set of nested equations. The syntax of the expressions follows the FORTRAN language standard except that all arguments are assumed to be real numbers (no integers). Intrinsic functions may also be used.

The equation input arguments are formal arguments that are defined at runtime. The DRESP2 entry for responses and the DVPREL2 entry for design variable-to-property relations specify the input, or actual, arguments to the appropriate DEQATN entry.

Note

If nested equations are used on a DEQATN entry, the return value is determined from the last equation on the entry.

A nested form of equation definition may also be used. Multiple equations can be specified on a single DEQATN entry with the input to the latter equations based on the results of the previous equations. The return value of the function call is the last equation in the DEQATN entry. The argument list, however, must contain all of the formal arguments used by all equations on the entry. For more details, see Relating Design Variables to Properties and Identifying the Design Responses .

DESVAR

Purpose:

Defines the design variables to be used in design sensitivity and optimization. Design sensitivity analysis computes the rates of change of design responses with respect to changes in the design variables. In design optimization, the set of design variables are the quantities modified by the optimizer in the search for an improved design.

Entry Description:

DESVAR	ID	LABEL	XINIT	XLB	XUB	DELXV	DDVAL	

Field	**Contents**
ID	Unique design variable identification number. (Integer > 0).
LABEL	User-supplied name for printing purposes. (Character).
XINIT	Initial value. (Real, XLB ≤ XINIT ≤ XUB).
XLB	Lower bound. (Real, default = -1 .0E+20).
XUB	Upper bound. (Real, default = +1 .0E+20).
DELXV	Move limit for the design variable during approximate optimization. (Real > 0.0).
DDVAL	ID of a DDVAL entry that provides a set of allowable discrete values. (Blank or Integer>0; Default=blank for continuous design variables.)

Associated Entries:

Design variables may be related to properties on DVPREL1 or DVPREL2 entries, related to changes in shape using DVBSHAP, DVGRID, or DVSHAP entries, linked using DLINK entries or input to user-defined responses on the DRESP2 entry.

DVPREL1	ID	TYPE	PID	FID	PMIN	PMAX	C0	
	DVID1	COEF1	DVID2	COEF2	DVID3	-etc.-		

DVPREL2	ID	TYPE	PID	FID	PMIN	PMAX	EQID	
	“DESVAR”	DVID1	DVID2	DVID3	-etc.-			
	“DTABLE”	LABL1	LABL2	LABL3	-etc.-			

DVBSHAP	DVID	AUXMOD	COL1	SF1	COL2	SF2	COL3	SF3

DVGRID	DVID	GID	CID	COEFF	N1	N2	N3	

DVSHAP	DVID	COL1	SF1	COL2	SF2	COL3	SF3	

DLINK	ID	DDVID	C0	CMULT	IDV1	C1	IDV2	C2
	IDV3	C3	-etc.-					

DRESP2	ID	LABEL	EQID	REGION				
	“DESVAR”	DVID1	DVID2	-etc.-				
	“DTABLE”	LABL1	LABL2	-etc.-				
	“DRESP1"	NR1	NR2	-etc.-				
	“DNODE”	G1	C1	G2	C2	-etc.-		

Discussion:

In design optimization, structural properties are changed in order to determine the optimal value of the objective function. To accomplish this, the optimizer varies the design variables. Not only must design variables be defined, but they also must be functionally related to analysis model properties and/or changes in structural shapes. The DESVAR entry is used to define an individual design variable. Design variables may be defined as members of independent and dependent sets using DLINK Bulk Data entries, if desired.

DLINK

Purpose:

Imposes a linear relationship among the design variables. The set of all DLINK entries partitions the design variables into an independent set and a dependent set.

Entry Description:

DLINK	ID	DDVID	CO	CMULT	IDV1	C1	IDV2	C2
	IDV3	C3	-etc.-					

Field	**Contents**
ID	Unique entry identifier. (Integer > 0).

Field	Contents
DDVID	Dependent design variable identification number. (Integer > 0).
C0	Constant term. (Real, default = 0.0).
CMULT	Constant multiplier. (Real, default = 1.0).
IDVi	Independent design variable identification number. (Integer > 0).
Ci	Coefficient i (corresponding to IDVi). (Real).

Associated Entries:

Design variables are identified in DLINK relations by their IDs (both independent and dependent variables).

DESVAR	ID	LABEL	XINIT	XLB	XUB	DELXV	DDVAL	

Discussion:

A DLINK entry specifies a design variable relationship of the form:

$$x_D = c_o + \sum_i c_i x_i$$

Equation 4-2.

where:

x_D	=	dependent design variable
c_o	=	constant
c_i	=	constant multiplying coefficient
x_i	=	independent design variables

Design variable linking can be used to ensure structural symmetry, unify sizing changes across property groups, etc. The efficiency of the design process is usually improved if the number of independent design variables can be kept to a minimum. Placing a design variable in the dependent set using a DLINK entry removes it from the independent set.

DOPTPRM

Purpose:

Overrides default values of parameters used in design optimization.

Entry Description:

DOPTPRM	PARAM1	VAL1	PARAM2	VAL2	PARAM3	VAL3	PARAM4	VAL4
	PARAM5	VAL5	-etc.-					

Field	Contents
PARAMi	Name of the design optimization parameter. For allowable names, see the DOPTPRM listing in Bulk Data Entries (Character).

Field	Contents
VALi	Value of the parameter. (Real or Integer, see the DOPTPRM listing in Bulk Data Entries).

Associated Entries:

There are no directly associated entries.

Discussion:

There are numerous parameters that control various aspects of the optimization process itself. While all of these parameters have defaults (the DOPTPRM entry is optional), the defaults can be changed using the DOPTPRM entry. See also the Bulk Data listing in Bulk Data Entries as it contains the complete list of all optimization parameters that may be changed. Following is an overview of the various parameters, grouped according to their functionality.

Choice of Approximation Method:

There are three types of approximation methods to choose from: direct linearization, mixed method, and convex linearization. The mixed method is the default.

Direct linearization (APRCOD = 1) is based on the simple first-order Taylor series expansion directly in terms of the design variables. The method is often useful for dynamic response optimization, shape optimization, and optimization tasks that use basis vector formulations.

The mixed method (APRCOD = 2) uses a combination of direct and reciprocal approximations, depending on the response type being approximated. Volume, weight, internal force, and buckling load responses are directly approximated, while all other response types use reciprocal approximations. APRCOD = 2 is the default.

Convex linearization (APRCOD = 3) uses either a direct or reciprocal approximation, depending on which one yields the more conservative approximation. The choice is made on an individual design variable and individual response-type basis.

For theoretical details, see Approximation Concepts in Design Optimization .

Print Controls:

Two types of print control are available. The first governs output at the design optimization cycle level, the second governs output at the optimizer level.

The first type of output control is provided by the parameters P1 and P2, which control the overall design cycle printout. P1 controls the frequency of output while P2 controls the quantity. Values for P1 and P2 are shown in the following tables:

P1	=0	output for initial and optimal designs (default)
	=n	output for every n-th design cycle

P2	=0	no output
	=1	objective and design variables (default)
	=2	designed properties
	=4	design constraints
	=8	design responses

The various P2 options can be summed to produce output combinations. For example, P2 = 15 prints out all available data.

The second type of output control is provided by the parameter IPRINT, which controls the amount of optimizer output. This parameter is normally turned off (default = 0) since the output generated upon exit from the optimizer (P1 and P2) is usually sufficient. However, sometimes it is useful to follow what is happening within the optimizer itself. The levels of optimizer print are as follows:

IPRINT	Optimizer Printout
0	No output (default)
1	Internal parameters, initial information, and results
2	Same, plus objective function and design variables at each iteration
3	Same, plus constraint values and identification of critical constraints
4	Same, plus gradients
5	Same, plus search direction
6	Same, plus scaling factors and miscellaneous search information
7	Same, plus one-dimensional search information

Maximum Number of Design Cycles:

The optimization process is iterative since the optimizer obtains data about the design space from approximations. The approximate model, constructed based on a detailed finite element analysis, is used by the optimizer to find an approximate optimum. This design is resubmitted for another finite element analysis followed by another approximate optimization. This process is repeated until convergence with respect to these overall design cycles is reached. The default maximum number of such cycles is five unless the problem converges before then. Usually, a near-optimum design is found by this point; however, more cycles are often necessary. If the maximum number of design cycles is reached before convergence is achieved, the problem can always be restarted from the last design (see Restarts in Design Optimization).

Move Limits on the Approximate Optimization:

As the optimizer modifies the design variables, the structure's properties and/or shape will vary depending on the design model description. As discussed in Section 3.3, Optimization with Respect to Approximate Models, move limits need to be placed on the approximate subproblem for efficiency reasons. These move limits are imposed with respect to analysis model properties as well as design variables. They can be changed from their defaults by modifying DELP and DPMIN for properties, and DELX, DELXV and DXMIN for design variables.

Convergence Criteria:

Convergence at the Design Cycle Level

The parameters CONV1, CONV2, GMAX, CONVDV, and CONVPR are used to test for overall design cycle convergence. These parameters are used in connection with tests for both hard and soft convergence. Convergence Tests , describes the types of convergence testing as well as the convergence decision logic.

Soft Convergence

Soft convergence compares the results of the approximate optimization with the results of the finite element analysis performed at the beginning of the design cycle. Soft convergence is not sufficient to terminate the design cycle iterations unless the parameter SOFTEXIT is set to YES. NO is the default.

Hard Convergence

Hard convergence testing compares the analysis results of current design cycle with those of the previous cycle. This test is a more conclusive test of convergence since it is based on hard evidence. Hard convergence will always terminate the design cycle process.

Convergence at the Optimizer Level

Convergence at the optimizer level can also be controlled using the DOPTPRM entry. Parameters that can be changed include DABOBJ, DELOBJ, ITMAX, ITRMOP, ITRMST, and JTMAX.

Finite Difference Step Sizes:

Design sensitivity coefficients are computed in NX Nastran using a semianalytic approach, as explained in Design Sensitivity Analysis .

For example, static displacement sensitivities are determined from the solution of the differentiated equilibrium equations

$$[K]\frac{\partial\{u\}}{\partial x_i} = \frac{\partial\{P\}}{\partial x_i} - \frac{\partial[K]}{\partial x_i}\{u\}$$

Equation 4-3.

Note

Central differences are available in NX Nastran. (In fact, central differences are the default for shape optimization.)

where $(\partial[K])/\partial x_i$ is approximated using finite differences as

$$\frac{\partial[K]}{\partial x_i} \cong \frac{[K(\vec{x}^0 + \mathrm{DELB}\cdot x_i)] - [K](\vec{x}^0)}{\mathrm{DELB}\cdot x_i}$$

Equation 4-4.

The value of the finite difference move parameter DELB can be changed with the DOPTPRM entry. If the product of DELB and x_i is small, the finite difference approximation may not be very accurate. A minimum finite move is then provided by DELBM, which is used if DELB · $x_i \leq$ DELBM.

Comparison Between the Initial and Calculated Values of Analysis Properties:

The initial value of an analysis property given on a property Bulk Data entry may not always correspond to the value of the property calculated using the initial values of the design variables. If the relative difference between the properties differ by more than PTOL, the program terminates with an error condition. The default of 1.0E35 implies that nearly all discrepancies are tolerated and the initial analysis property values will be overridden using the design model data.

DRESP1

Purpose:

Defines direct, or first-level, analysis responses to be used in design sensitivity and optimization. The responses identified here are those that are directly available from the analysis results as opposed to second-level responses that are defined using DVPREL2 and DEQATN entries.

Entry Description:

DRESP1	ID	LABEL	RTYPE	PTYPE	REGION	ATTA	ATTB	ATT1
	ATT2	-etc.-						

Field	Contents
ID	Unique entry identifier. (Integer > 0).
LABEL	User-defined label. (Character).
RTYPE	Response type. See DRESP1 listing in Commonly Used Commands for Design Optimization . (Character).
PTYPE	Element flag (PTYPE = “ELEM”) or property entry name. Used with element type responses (stress, strain, force, etc.) to identify the property type, since property entry IDs are not unique across property types. (Character: “ELEM,” “PBAR,” “PSHELL,” etc.).
REGION	Region identifier for constraint screening. (Integer > 0).
ATTA, ATTB, ATTi	Response attributes. See DRESP1 listing in Commonly Used Commands for Design Optimization . (Integer > 0 or Real or blank).

Associated Entries:

A response identified on a DRESP1 entry may be used either as a constraint or an objective. The response is identified by its ID number on DCONSTR entries.

DCONSTR	DCID	RID	LALLOW	UALLOW				

A DRESP1 response can also be used as input to synthetic responses defined on DRESP2 entries. It is referenced by its response ID.

DRESP2	ID	LABEL	EQID	REGION					
	“DESVAR”	DVID1	DVID2	-etc.-					
	“DTABLE”	LABL1	LABL2	-etc.-					
	“DRESP1"	NR1	NR2	-etc.-					
	“DNODE”	G1	C1	G2	C2	-etc.-			

A first-level response can also be used as an objective if it defines a single, scalar response (e.g., weight, an eigenvalue, a grid displacement component for a single subcase, and so on). The objective is defined in Case Control with the command

$$DESOBJ\begin{pmatrix} min \\ max \end{pmatrix} = n$$

Equation 4-5.

where n is the DRESP1 (or DRESP2) entry ID.

Discussion:

Depending on the particular analysis discipline, responses such as displacements, stresses, eigenvalues, etc., may be computed. However, not all of these responses may be of interest from a design optimization standpoint. The DRESP1 entry is used to identify responses that are to be used in connection with design sensitivity and optimization either as an objective or as a constraint. These responses may also be used as input to compute a second-level, or synthetic, response on a DRESP2 entry.

An internal Case Control for design sensitivity and optimization is built using the set of first-level responses. Selecting design responses using DRESP1 entries also ensures that the appropriate data recovery is performed, regardless of Case Control requests the user may have supplied. Output requests, if desired, still must be specified with the appropriate Case Control commands; however, this is for postprocessing convenience only and is not a requirement for either design sensitivity or optimization.

DRESP2

Purpose:

Defines, in conjunction with a DEQATN entry, a second-level response to be used either as an objective or as a constraint.

Entry Description:

DRESP2	ID	LABEL	EQID	REGION					
	“DESVAR”	DVID1	DVID2	-etc.-					
	“DTABLE”	LABL1	LABL2	-etc.-					
	“DRESP1"	NR1	NR2	-etc.-					
	“DNODE”	G1	C1	G2	C2	-etc.-			

Field	Contents
ID	Unique identification number. (Integer > 0).
LABEL	User-defined label. (Character).
EQID	DEQATN entry identification number. (Integer > 0).
REGION	Region identifier for constraint screening. (Integer > 0).
“DESVAR”	Flag indicating DESVAR entry identification numbers. (Character).
DVIDi	DESVAR entry identification number. (Integer > 0).
“DTABLE”	Flag indicating that the labels for the constants in a DTABLE entry follow. (Character).
LABLj	Label for a constant in the DTABLE entry. (Character).

Field	Contents
“DRESP1"	Flag indicating DRESP1 entry identification numbers. (Character).
NRk	DRESP1 entry identification number. (Integer > 0).
“DNODE”	Flag signifying that the following fields are designed grid points. (Character).
Gm	Grid point identification number. (Integer > 0).
Cm	Degree of freedom number of grid point Gm. (1 ≤ Integer ≤ 3).

Associated Entries:

The DCONSTR entry can be used to place bounds on a DRESP2 response using the DRESP2 entry ID as a reference:

DCONSTR	DCID	RID	LALLOW	UALLOW					

The DESOBJ Case Control command can also reference the DRESP2 entry ID:

$$DESOBJ\begin{pmatrix} min \\ max \end{pmatrix} = n$$

Equation 4-6.

Discussion:

It is often necessary to define structural responses that NX Nastran does not calculate directly, e.g., local buckling criteria. The response equation is written on a DEQATN entry. This type of response is referred to as “second-level” as opposed to “first-level” responses that are directly available from the analysis.

The DRESP2 entry defines the input, or actual, arguments to these equations. The arguments may be design variables, table constants, first-level structural responses, or grid coordinate locations.

DSCREEN

Purpose:

Defines override information necessary to screen the constraints for temporary deletion.

Entry Description:

DSCREEN	RTYPE	TRS	NSTR						

Field	Contents
RTYPE	Response type for which the screening criteria apply. (Character).
TRS	Truncation threshold. (Real; Default = -0.5).
NSTR	Maximum number of constraints to be retained per region per load case. (Integer > 0; Default = 20).

Associated Entries:

The regions for constraint screening are automatically established by default. These defaults can be overridden though and new ones can be established by defining new regions on the DRESP1 and DRESP2 entries.

DRESP1	ID	LABEL	RTYPE	PTYPE	REGION	ATTA	ATTB	ATT1
	ATT2	-etc.-						

DRESP2	ID	LABEL	EQID	REGION				
	“DESVAR”	DVID1	DVID2	-etc.-				
	“DTABLE”	LABL1	LABL2	-etc.-				
	“DRESP1"	NR1	NR2	-etc.-				
	“DNODE”	G1	C1	G2	C2	-etc.-		

Discussion:

To reduce the costs associated with the sensitivity analysis and to reduce the size of the optimization problem, many of the noncritical constraints are dynamically deleted during the optimization process. Default values of parameters that control the screening process can be overridden using DSCREEN entries. Since these defaults are often satisfactory, including DSCREEN entries is often unnecessary.

Constraint Deletion

Constraint screening is performed in two stages. In the first step, deletion, the constraint values that fall below a certain threshold level are deleted from the constraint set. The default threshold level is –0.5, but this can be overridden by specifying a TRS value on the DSCREEN entry. In NX Nastran, the normalized constraints are considered violated if they are positive quantities. The default TRS states that only those constraints that are within 50% of their critical values are retained.

Constraint Regionalization

The second step, regionalization, sorts the remaining constraints and retains up to a prescribed maximum per region per load case. The default region specifications are given on the DSCREEN Bulk Data entry description. NSTR, or the number of constraints to be retained per region, has a default value of 20, which again can be overridden.

DTABLE

Purpose:

Defines a table of constants to be used in conjunction with DEQATN equations.

Entry Description:

DTABLE	LABL1	VALU1	LABL2	VALU2	LABL3	VALU3	LABL4	VALU4
	LABL5	VALU5	LABL6	VALU6	-etc.-			

Field	Contents
LABLi	Label for the constant. (Character).
VALUi	Value of the constant. (Real).

Associated Entries:

DRESP2 and DVPREL2 entries list the input arguments to equations that define synthetic response and property relations, respectively. These input arguments may include table constants defined on the DTABLE entry.

DRESP2	ID	LABEL	EQID	REGION				
	"DESVAR"	DVID1	DVID2	-etc.-				
	"DTABLE"	LABL1	LABL2	-etc.-				
	"DRESP1"	NR1	NR2	-etc.-				
	"DNODE"	G1	C1	G2	C2	-etc.-		

DVPREL2	ID	TYPE	PID	FID	PMIN	PMAX	EQID	
	"DESVAR"	DVID1	DVID2	DVID3	-etc.-			
	"DTABLE"	LABL1	LABL2	LABL3	-etc.-			

Discussion:

Constants used in equations can either be "built into" the DEQATN entry when the equation is defined or passed as arguments. Building-in the values of constants may be inconvenient and can prevent one equation from easily being used in different contexts.

The DTABLE entry allows these constants to be stored in a table and then used in the equations as necessary. On the DTABLE entry, a constant is given a name and a value. When the equation argument list is defined on a DVPREL2 or DRESP2 entry, the constant is referenced by its name.

Only one DTABLE entry may appear in the Bulk Data.

DVBSHAP

Purpose:

Defines a shape basis vector as a linear combination of analytic boundary shapes solutions and assigns a design variable to the result.

Entry Description:

DVBSHAP	DVID	AUXMOD	COL1	SF1	COL2	SF2	COL3	SF3

Field	Contents
DVID	Design variable identification number of a DESVAR entry. (Integer > 0).
AUXMOD	Auxiliary model identification number. (Integer > 0).
COLi	Load sequence identification number from AUXMODEL Case Control command. (Integer > 0).
SFi	Scaling factor for load sequence identification number. (Real; Default = 1.0).

Associated Entries:

In shape optimization, shape basis vectors relate the changes in a design variable to changes in grid locations (see basis vectors in shape optimization, Relating Variables to Shape Changes.) Design variables must first be defined using DESVAR entries.

DESVAR	ID	LABEL	XINIT	XLB	XUB	DELXV	DDVAL	

DVGRID

Purpose:

Defines design variable-to-grid coordinate relations for shape sensitivity and optimization.

Entry Description:

DVGRID	DVID	GID	CID	COEFF	N1	N2	N3	

Field	Contents
DVID	DESVAR entry identification number. (Integer > 0).
GID	Grid point identification number. (Integer > 0).
CID	Coordinate system identification number. (Integer ≥ 0; Default = 0).
COEFF	Multiplier of the vector defined by Ni. (Real; Default = 0.0).
Ni	Components of the vector measured in the coordinate system defined by CID. (Real; at least one Ni ≠ 0.0).

Associated Entries:

The DESVAR entry defines a design variable that can be used to describe grid variations as well as other design relations. The “DNODE” fields on the DRESP2 entry allow the coordinates of designed grids to be used in formulating second-level responses.

DESVAR	ID	LABEL	XINIT	XLB	XUB	DELXV	DDVAL	

DRESP2	ID	LABEL	EQID	REGION				
	“DESVAR”	DVID1	DVID2	-etc.-				
	“DTABLE”	LABL1	LABL2	-etc.-				
	“DRESP1"	NR1	NR2	-etc.-				
	“DNODE”	G1	C1	G2	C2	-etc.-		

Discussion:

Changes in grid coordinate values are expressed as functions of design variables according to the relation

$$\{\Delta G_i\} = \sum_i \{T\}_{ij} \cdot \Delta x_j$$

Equation 4-7.

where changes in the *i*-th grid point are expressed in terms of changes in the *j*-th design variable. If the number of design variables is less than the number of grid points, this relation is called a reduced basis formulation. Reduced basis formulations are discussed in Relating Design Variables to Shape Changes .

The DVGRID entry defines the particular grid point, design variable, and components of each $\{T\}_{ij}$ vector. Multiple references to the same grid point and design variable pair simply result in vectorial addition of the corresponding $\{T\}_{ij}$.

Other shape basis vector definition methods utilize DVBSHAP and DVSHAP Bulk Data entries.

DVPREL1

Purpose:

Defines a structural analysis property as a function of a linear combination of design variables.

Entry Description:

DVPREL1	ID	TYPE	PID	FID	PMIN	PMAX	C0	
	DVID1	COEF1	DVID2	COEF2	DVID3	-etc.-		

Field	Contents
ID	Unique identification number. (Integer > 0).
TYPE	Name of a property entry, such as “PBAR”, “PBEAM”, etc. (Character).
PID	Property entry identification number. (Integer > 0).
FID	Field position of the property entry, or word position in the element property table of the analysis model. (Integer ≠ 0.0).
PMIN	Minimum value allowed for this property. If FID references a stress recovery location, then the default value for PMIN is -1.0+35. PMIN must be explicitly set to a negative number for properties that may be less than zero (for example, field ZO on the PCOMP entry). (Real; Default = 0.001).
PMAX	Maximum value allowed for this property. (Real; Default = 1.0E20).
C0	Constant term of relation. (Real; Default = 0.0).
DVIDi	DESVAR entry identification number. (Integer > 0).
COEFi	Coefficient of linear relation. (Real).

Associated Entries:

Design variables are referenced on DVPREL1 entries by their DESVAR-defined IDs.

DESVAR	ID	LABEL	XINIT	XLB	XUB	DELXV	DDVAL	

Discussion:

An analysis model property can be expressed as a linear combination of design variables as

$$p_1 = c_0 + c_1 x_1 + c_2 x_2 + \dots + c_n x_n$$

Equation 4-8.

Both independent as well as dependent design variables can appear in this relation (see the DLINK entry). This form is especially useful in the sense that it can be used to express not only a simple one-to-one correspondence between a design variable and a property, but more complex reduced basis formulations as well.

DVPREL2

Purpose:

Defines a structural property by reference to an equation defined on a DEQATN entry.

Entry Description:

DVPREL2	ID	TYPE	PID	FID	PMIN	PMAX	EQID	
	"DESVAR"	DVID1	DVID2	-etc.-				
	"DTABLE"	LABL1	LABL2	-etc.-				

Field	Contents
ID	Unique identification number. (Integer > 0).
TYPE	Name of a property entry, such as PBAR, PBEAM, etc. (Character).
PID	Property entry identification number. (Integer > 0).
FID	Field position of the property in the analysis model entry. (Integer ≠ 0.0).
PMIN	Minimum value allowed for this property. If FID references a stress recovery location field, then the default value for PMIN is -1.0+35. PMIN must be explicitly set to a negative number for properties that may be less than zero (for example, field ZO on the PCOMP entry). (Real; Default = 0.001).
PMAX	Maximum value allowed for this property. (Real; Default = 1.0E20).
EQID	DEQATN entry identification number. (Integer > 0).
"DESVAR"	DESVAR flag. Indicates that the IDs of DESVAR entries follow. (Character).
DVIDi	DESVAR entry identification number. (Integer > 0).
"DTABLE"	DTABLE flag. Indicates that the IDs for the constants in a DTABLE entry follow. This field may be omitted if there are no constants involved in this relation. (Character).
LABi	Label for a constant on the DTABLE entry. (Integer > 0).

Associated Entries:

To define a synthetic property relation, the DVPREL2 entry identifies a DEQATN entry and declares the design variable (DESVAR) and any necessary table constant (DTABLE) arguments.

DEQATN	EQID	EQUATION
	EQUATION (Cont.)	

DESVAR	ID	LABEL	XINIT	XLB	XUB	DELXV	DDVAL	

DTABLE	LABL1	VALU1	LABL2	VALU2	LABL3	VALU3	LABL4	VALU4
	LABL5	VALU5	LABL6	VALU6	-etc.-			

Discussion:

For many applications, the linear design variable-to-property relations provided by the DVPREL1 entry may not be sufficient. In these cases, equations may be defined to relate design variables to properties in a nonlinear fashion. The DVPREL2 entry provides the arguments to an equation defined on a DEQATN entry. These arguments may be design variables defined on DESVAR entries and table constants defined on a DTABLE entry.

Design Sensitivities and Type-2 Properties

Every property defined on a DVPREL2 entry becomes an independently varying property internally in NX Nastran. Design sensitivities are first computed with respect to these independently varying properties, and then the chain rule applied to relate these sensitivities to changes in the design variables. Refer to Design Sensitivity Analysis , for further details.

DVSHAP

Purpose:

Defines a shape basis vector as a linear combination of DBLOCATEd displacement vectors (in data block UGD) and assigns a design variable multiplier to the result.

Entry Description:

DVSHAP	DVID	COL1	SF1	COL2	SF2	COL3	SF3	

Field	Contents
DVID	Design variable identification number on the DESVAR entry. (Integer > 0).
COLi	Column number of the displacement matrix. (1 ≤ Integer ≤ maximum column number in the displacement matrix).
SFi	Scaling factor applied to the COLi-th column of the displacement matrix. (Real; Default = 1.0).

Associated Entries:

In shape optimization, shape basis vectors relate the changes in a design variable to changes in grid locations (see basis vectors in shape optimization in Relating Design Variables to Shape Changes). The design variables must first be defined using DESVAR entries.

DESVAR	ID	LABEL	XINIT	XLB	XUB	DELXV	DDVAL	

4.4 Parameters for Design Sensitivity and Optimization

In addition to the optimization control possible with Bulk Data entries such as DOPTPRM and DSCREEN, a set of parameters unique to design sensitivity and optimization is also available. This section is intended to be a complete quick reference for all such Case Control and Bulk Data parameters. Refer to Commonly Used Commands for Design Optimization for further details.

CDIF	Controls the selection of finite difference scheme used in sensitivity analysis. YES: Selects central differences (default for shape optimization). NO: Selects forward differences (default for property optimization only)

DESPCH	Controls the frequency of updated DESVAR and GRID Bulk Data entry output to the PUNCH file. N < 0: No output. N = 0: Final design cycle (default). N > 0: Every N-th, as well as final design cycles.
DSNOKD	Indicates if the differential stiffness effect is to be included in buckling sensitivity analysis. 1.0: Include differential stiffness (default). 0.0: Do not include differential stiffness.
NASPRT	Controls the frequency of NX Nastran output. -1: No output. 0: Output on initial and final design cycles (default). N: Output every N-th iteration and also prior to exit.
OPTEXIT	Instructs the program to exit at one of six predetermined exit points. 0: No user-defined exit (default). N > 0: Exit at one of the locations 1 through 7 ($1 \leq N \leq 7$).
SOFTEXIT	Determines whether to terminate design cycles if soft convergence is indicated. NO: Do not stop if soft convergence is indicated (default). YES: Terminate design cycles if soft convergence is achieved.
UPDTBSH	Controls the update of the boundary shapes in the analytic boundary shapes method for shape optimization. NO: Do not update the boundary shapes (default). YES: Update the boundary shapes. **Note** Regardless of the value of UPDTBSH, shape basis vectors are still updated for every design cycle (interpolation to the interior

Chapter 5: Solution Sequences

For most applications, it is unlikely that an extensive understanding of the design optimization modules and data blocks will be required. However, a general knowledge of the module functions may be of help when evaluating design optimization messages, user warning messages, or error messages. An awareness of the contents of some of the more significant data blocks may also be useful when examining intermediate results, debugging input data, etc.

This section briefly outlines the function of those modules that are unique to design optimization. Most of the module names begin with the letters DO (which stands for design optimization) or DS (for design sensitivity). If necessary, the *NX Nastran Programmer's Manual* should be consulted in connection with advanced applications.

Following the module and data block descriptions is a top-level flowchart of Solution Sequence 200. This flowchart is referenced in Output Features and Interpretation and Example Problems in connection with output interpretation.

5.1 Design Sensitivity and Optimization Modules

AXMDRV	(Auxiliary model driver) drives the auxiliary model looping (for shape optimization using auxiliary boundary models) by reading the auxiliary model list and outputting the next sequential auxiliary model ID (AUXMID). The auxiliary model loop flag AMLPFL is FALSE if this is the last auxiliary model in the list.
AXMPR1	Builds the auxiliary model list AMLIST used in the auxiliary model boundary shapes method for shape optimization.
AXMPR2	Generates complete geometry (grid) data for the auxiliary boundary models by merging with the geometry of the primary structure. It also generates a Case Control data block CASEVEC used later to generate a geometry partitioning vector. This is used if additional grids have been defined by the engineer in the auxiliary boundary model.
DMPR	Partitions the design model entries table (EDOM) on analysis type. This is the first of two partitioning operations; the second step, partitioning on superelements, is performed later by SDSA. Thus, for all later data processing operations, which usually involve looping over analysis disciplines and superelements, we will have on hand only that required subset of design model entries.
DOM6	Computes the design sensitivity coefficient matrix DSCM2 for output purposes. This module is invoked only if OPTEXIT is + 4. Otherwise, design sensitivity information is available to the optimizer via matrix DSCM.
DOM9	Performs the optimization task using the previously created approximating functions.
DOM10	Provides printed output of the results of the approximate optimization. This output includes the values of the proposed new design variables and the estimated values of the objective and constraint functions. The amount of output is determined by the parameters P1 and P2 on the DOPTPRM Bulk Data entry.
DOM11	Updates the element property table (EPT), grid coordinates (COORDO), and geometry data blocks (GEOM1N) in accordance with the new vector of design variables.
DOM12	Performs both hard and soft convergence testing, outputs the results of these tests, updates the design optimization history table, and prints out the summary of design cycle history. It is called for every normal exit condition in Solution 200.
DOPR1	Sets up nonrepetetive tables for design variables, design variable linking, DTABLE constants, designed property attributes, initial design variable and property values, constraint screening data, and optimization parameters.

DOPR2	Performs pre-processing operations for the shape basis vectors (sensitivity and optimization). It is called in PREDOM for direct input of basis vectors, and DESOPT for the AMBS and GMBS options. Creates the shape basis vectors in basic coordinates as well as the necessary design variable correlation data.
DOPR3	Generates tables that are used to compute both first- and second-level responses, constraints, and the objective function. Since these tables are invariant, they are formed on the first design cycle.
DOPR4	Expands tables DTOS2J and DTOS4J to account for DVPREL2 properties, which are treated as internal design variables. Design variable numbering in the resultant DTOS2K and DTOS4K tables is in terms of the external design variables (DESVAR input) and the internal (DVPREL2) variables.
DOPR5	Generates updated DTOS2 and DTOS4 tables from DTOS2K and DTOS4K using the new vector of design variables XINIT.
DSABO	In the semianalytic sensitivity analysis approach used in NX Nastran, perturbed structural matrices must be generated. DSABO constructs tables to which later modules refer when building property-related variational terms. These tables are primarily in the form of perturbed element summary tables and design sensitivity processing tables.
DSAD	Evaluates and screens constraints that are functions of either first- or second-level responses.
DSAE	For purposes of computational efficiency, the design variables are split into two sets. The first set consists of those design variables that are related to structural properties, while the second set consists of variables related to grid coordinates. DSAE merges the tables and data blocks associated with these groups of design variables so that later modules can compute the perturbed structural configurations.
DSAF	Generates an updated element summary table (ESTDCN) for recovery of responses in design sensitivity analysis. It contains the subset of elements necessary to recover element-level responses and incorporates the effects of constraint deletion.
DSAH	Forms a number of miscellaneous tables and data blocks, including Case Control that are used in the formation of the design response sensitivities.
DSAJ	Converts grid variations defined on the boundaries of the structure (geometry model boundary solution for shape optimization) to a shape basis vector. This in turn is used to define enforced displacements on the primary structure, for purposes of interpolation to the interior grids of the mesh.
DSAL	Actually computes the matrix of partial sensitivity coefficient DSCM given the baseline and perturbed responses, and corresponding design variable data. The form of the terms in DSCM is Δr/DELB. This form is converted to $\Delta r/\Delta x$ in modules DOM6 and DOM9.
DSAM	Same concept as the DSABO module but with respect to structural shape variations.
DSAN	Generates Design Sensitivity Table 1 DSPT1 used in DSVG1 and DSVG2 to generate the pseudo-load vectors.
DSAP	In direct and modal frequency analysis, DSAP sums the variational terms that are functions of ΔK, ΔB and ΔM to form the pseudo-load vector.
DSAR	Splits the modal transient solution containing data for displacement, velocity, and acceleration onto separate data blocks: one data block each for displacement, velocity, and acceleration for all solution time steps.
DSAW	Computes the delta weight and/or volume according to the changes in the design variables in connection with the semi-analytic sensitivity analysis.

DSPRM	Outputs parameters used for DMAP flow control. Depending on the types of retained responses listed in data block DRSTBL, parameters are output to indicate whether or not response sensitivities are to be recovered for a particular superelement.
DSTAP2	Generates a correlation table describing the design response column order for the DSCM/DSCM2 matrices.
DSVG1	(Design Sensitivity Vector Generator 1) computes terms in the sensitivity analysis pseudo-load vectors that depend on ΔK, ΔB, and ΔM.
DSVG2	(Design Sensitivity Vector Generator 2) This module computes the thermal load contribution to the pseudo-load vector.
DSVG4	(Design Sensitivity Vector Generator 4) Forms $u + \Delta u$ finite difference change in displacements for design variables that affect mass and/or stiffness terms.
MDCASE	Partitions the original input case control onto individual Case Control records based on analysis type. It also outputs parameters that are used at the DMAP level to direct analysis for the various disciplines.
SDSA	Partitions the design model entries table (EDOM) on superelement. The result, EDOMS, is an analysis discipline and superelement partition of the original EDOM. See also the DMPR module description.
SDSB	Builds the superelement design sensitivity processing data block DSLIST that is used to direct the pseudo-load and response sensitivity superement loops. Its structure is similar to SLIST, used in superelement analysis.
SDSC	Prints the DSCMR correlation table that identifies the column-response correlation in the design sensitivity coefficient matrix DSCM2.
SHPCAS	Generates a new Case Control data block to drive solutions for basis vectors in shape optimization. The initial shape data is supplied as enforced displacements on the boundary of the structure and then interpolated to the interior grids via a static solution of the primary structure. SHPCAS generates the Case Control data block CASENEW, which is used in connection with this solution. It also generates a partitioning vector CVEC to extract the appropriate solution vectors from the analysis results.
WEIGHT	Computes the structural weight or volume for use in design optimization. Weight (or mass) is only computed for those elements that have volume. The effect of elements such as CONMs are thus neglected.

5.2 Selected Data Blocks

The data blocks listed in this section are of particular interest in design sensitivity and optimization. Although just a subset, the data blocks listed here have been chosen based on their usefulness in custom DMAPing, results processing, etc.

Should the need arise, the brief descriptions in this section may be a useful aid when trying to navigate through various sections of the DMAP. The module that creates each data block is noted at the end of each description. Full descriptions of these data blocks' content and format can be found in the *NX Nastran Programmer's Manual*.

Design Model Definition

CVALO	Updated approximate constraint values output from the optimizer. (DOM9)
DESTAB	Table containing design variable attributes. (DOPR1)

DTOS2	Table, built from DTOS2K, containing current values of designed properties. This is updated at the beginning of every design cycle. (DOPR5)
DTOS2J	Table identifying properties referenced in the design model, both DVPREL1 and DVPREL2. This is built directly from the design model data. (DOPR1)
DTOS2K	Same as DTOS2J, but rearranged on internal design variable order. (DOPR4)
EDOM	Table containing design model data. (IFP)
EDOMS	Analysis discipline- and superelement-partitioned EDOM. (SDSA)
EPTN	Updated element property tables, based on optimization results. (DOM11)
GEOM1N	Updated grid (geometry) records, based on the optimization results. (DOM11)
PROPO	Vector of properties output from the optimizer. (DOM9)
R1VALO	First-level responses output from the optimizer. These are based on the approximate model. (DOM9)
R2VALO	Second-level responses output from the optimizer. These are based on the approximate model. (DOM9)
TABDEQ	Table of unique design variable IDs. (DOPR4)
XINIT	Vector of initial design variable values, updated at the start of each design cycle. (DOPR1 initialization, EQUIVX after DOM9 on subsequent cycles.)
XO	Vector of design variables output from the optimizer. This becomes XINIT for the next design cycle. (DOM9)

Shape Optimization

AMLIST	List of auxiliary models that is used to drive auxiliary model loops for shape optimization. (AXMPR1)
BASVEC	Basis vectors for shape optimization. This form is G-set size and is in basic coordinates. It is used in the AMBS, GMBS, and external input of basis vectors methods. (VECPLOT for AMBS and GMBS, SMPYAD for External Input.)
COORDO	Old set of coordinate values, for designed grids only. It is constant over all design cycles for the external input of basis vectors method and is updated on every design cycle for AMBS and GMBS. (DOPR2)
COORDN	New set of coordinate values for designed grids only. Its purpose is to provide the DNODE input data for the DRESP2 Bulk Data entries. (DOM11)
CVEC	Partitioning vector used to strip out the shape basis vector solutions for the AMBS and GMBS methods. (SHPCAS)
DELBSH	A vector defining the finite difference moves to be used in shape sensitivity. It is computed based on shape basis vector energy norms. (ADD5)
DESGID	List of designed coordinates, and identification data. (DOPR2)
DTOS4	Table containing designed grid perturbations. Its form is the same as DTOS4K and is used to define element-level variations for shape sensitivity analysis. (DOPR5)
DTOS4J	Table of shape basis vector data. It is in external (user-defined) design variable ID order. (DOPR2)
DTOS4K	Differs from DTOS4J only in that it is sorted on internal design variable ID. (DOPR4)
DVIDS	Design variables related to shape changes. (DSAJ)
DESVCP	Basis vectors for shape optimization, G-set size, expressed in the global coordinate system. DESVECP contains one column for each independent design variable. It is generated primarily for energy norm based step size determination. (DOPR2)
SHPVEC	Basis vectors for shape optimization, although it contains only designed grids. It is expressed in global coordinates with one column for each independent design variable. (DOPR2)

UGBSHP	G-set size basis vectors in the global coordinate system. It contains information for all grids in the model. (PARTN)

Constraint Evaluation and Screening

CASEDS	Case Control for the recovery of design responses. (DOPR3)
DRSTBL	Table providing the number of retained responses for each subcase for each of the response types. (DSAD)
R1VAL	Vector of type 1 response values. (DSAD)
R2VAL	Vector of type 2 response values. (DSAD)
R3VAL	Vector of type 3 response values (DSAD)
R1VALR	Vector of retained type 1 response values. (DSAD)
R2VALR	Vector of retained type 2 response values. (DSAD)
R3VALR	Vector of retained type 3 response values. (DSAD)
CVAL	Vector of constraint values. (DSAD)
CVALR	Vector of retained constraint values. (DSAD)

Design Sensitivity Analysis

DELWS	Delta weight. (DSAW)
DELVS	Delta volume. (DSAW)
DSCM	Design sensitivity coefficient matrix, form 1. Each term is Δr/DELB. (DSAL)
DSCM2	Design sensitivity coefficient matrix, form 2 (generated for output purposes only). Each term is $\Delta r/\Delta x$. (DOM6)
DSCMCOL	Correlation table for DSCM. (DSTAP2, SDSC)
ESTDVP	Perturbed element summary table used to compute property sensitivities. (DSABO)
ESTDVS	Perturbed element summary table used to compute shape sensitivities. (DSAM)
EGK	Pseudo-load contributions from changes in K. (DSVG1)
EGM	Pseudo-load contributions from changes in M. (DSVG1)
EGB	Pseudo-load contributions from changes in B. (DSVG1)

Design Optimization History

HIS	Design optimization history table. (DOM12→APPEND→EQUIVX)
HISADD	Additions to the design optimization history table due to the current design cycle. (DOM12)
NEWPRM	Update of design optimization parameters (due to the automatic reduction of move limits). (DOM12)

General

DSLIST	Superelement processing list for design optimization-similar to an SLIST for superelement analysis. (SDSB)
DTB	Constants from the DTABLE Bulk Data entry. (DOPR1)
OPTPRM	Optimization parameters. (DOPR1)

5.3 Solution 200 Program Flow

Design sensitivity analysis and optimization both are only available in Solution Sequence 200. The following flowchart describes the organization of Solution 200 or main subDMAP DESOPT and lists some of the more significant modules and subDMAPs invoked.

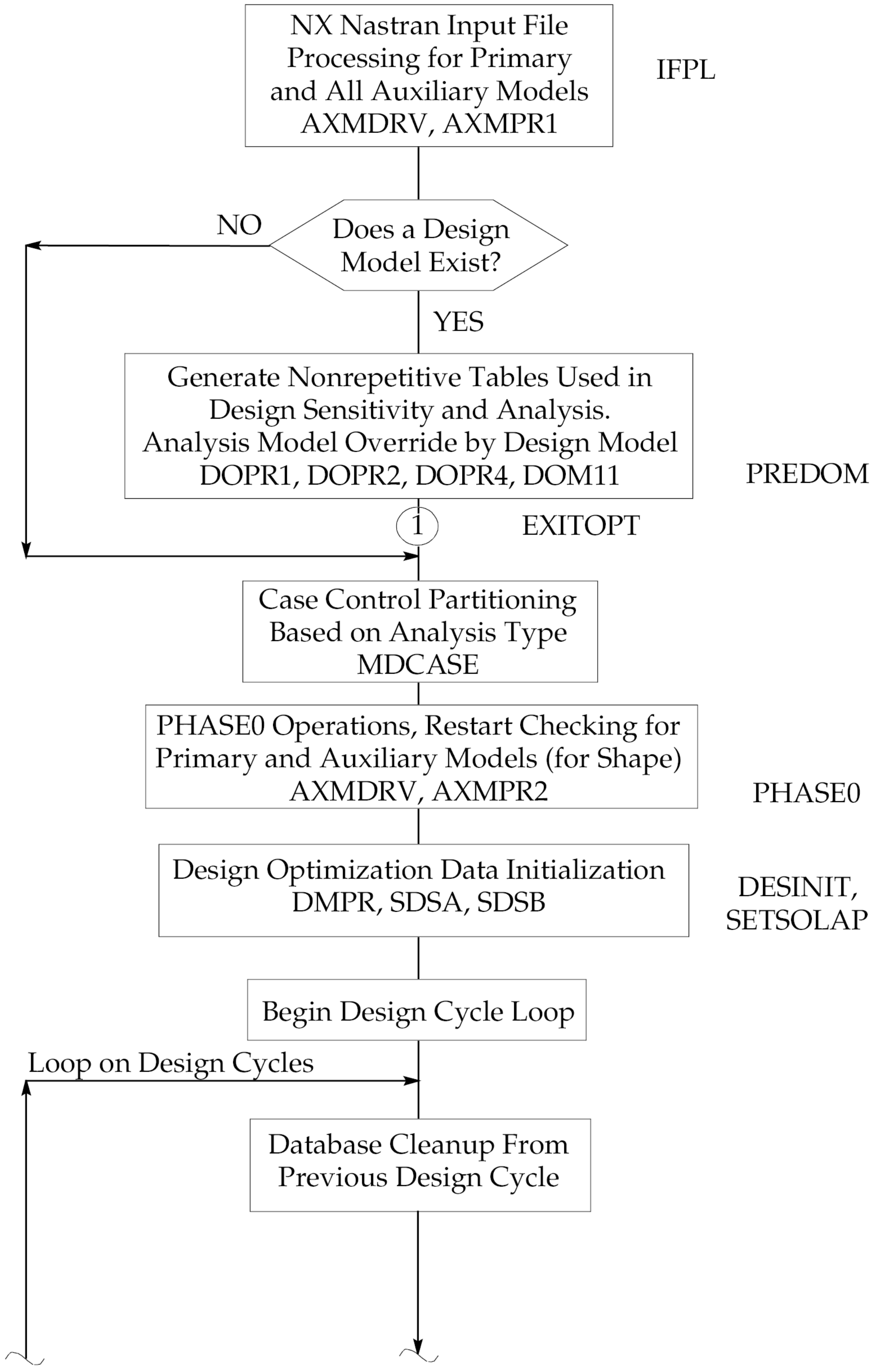
Each of the boxes in this high-level flowchart describes a major program phase as well as the design sensitivity and/or optimization modules used to accomplish that particular task. These module names are included inside each of the boxes. If program flow is directed to a particular subDMAP, each name is listed alongside each of the boxes.

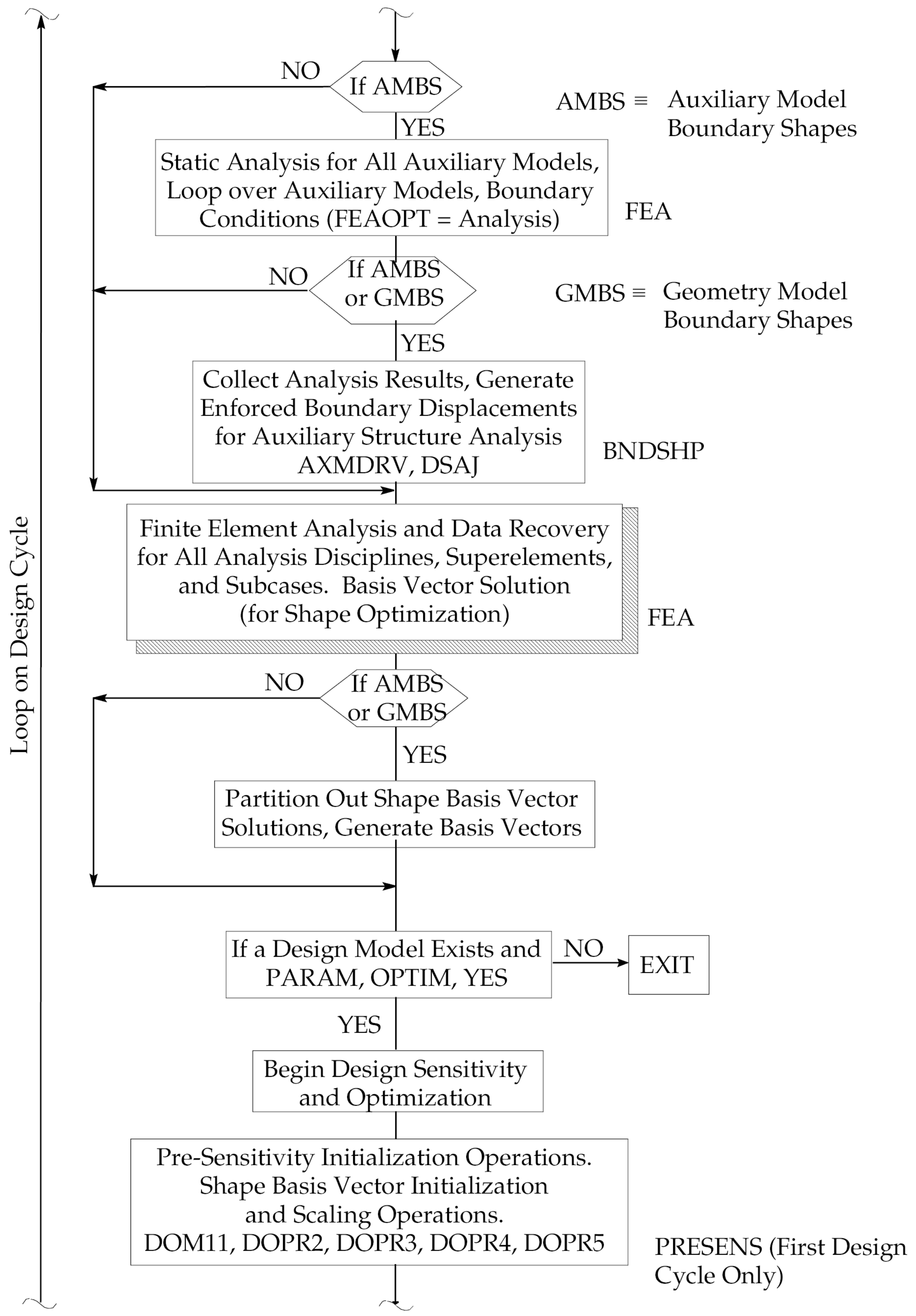
Note

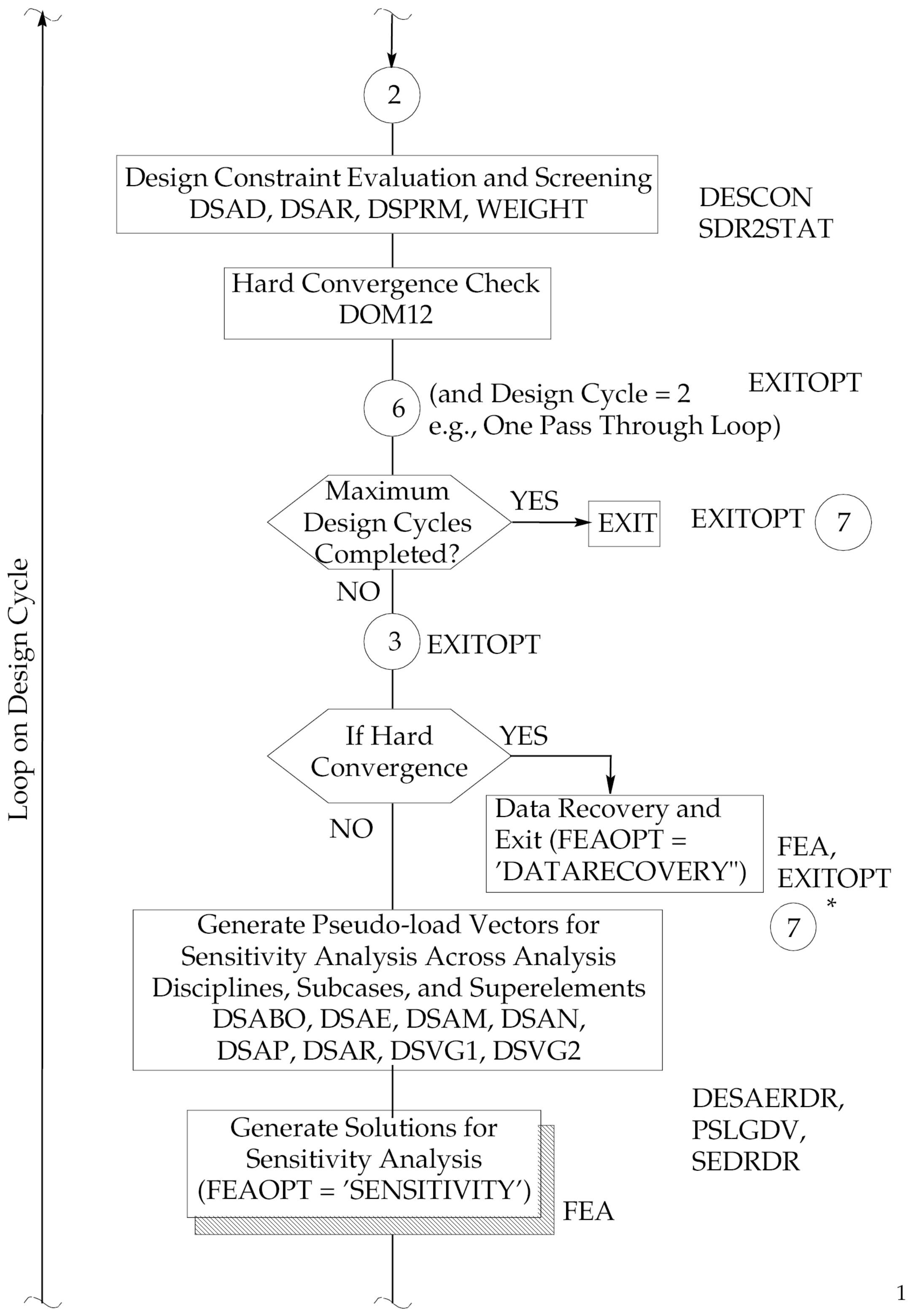
Finite element analysis and sensitivity analysis are the most computationally-intensive phases. These CPU-intensive phases have been highlighted in the flowchart.

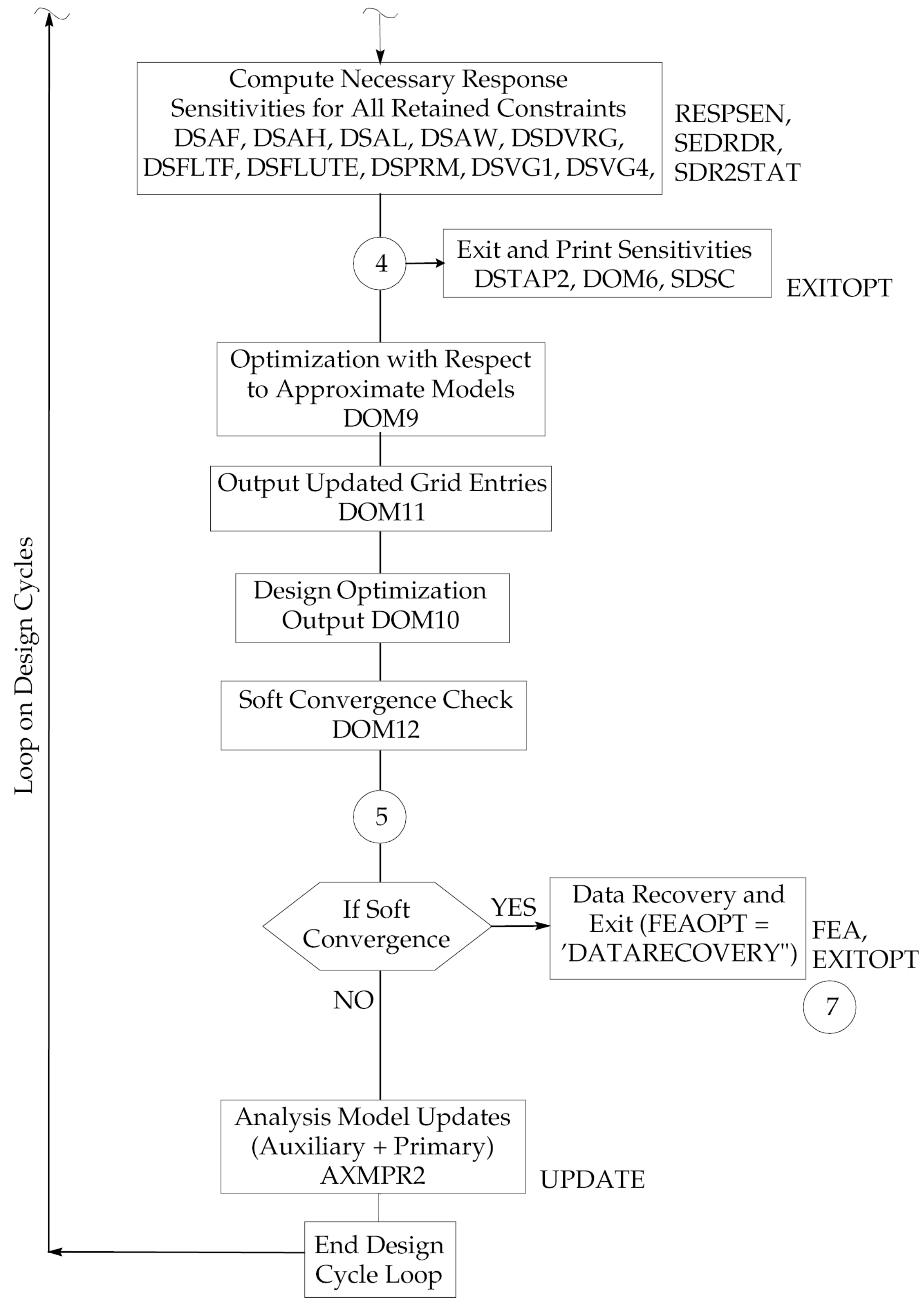
The circled numbers 1 through 6 appearing in the flowchart indicate the locations of the six user-defined exit points. Any one of these exits can be taken by setting the parameter OPTEXIT to an integer value from 1 through 6.

An OPTEXIT value of 7 provides sensitivity coefficient output upon normal program termination (the sensitivities are printed to the .f06 output file.) Normal program termination includes both hard and soft convergence, and maximum number of design cycles. Sensitivities for the final design may be particularly useful depending on your postprocessing requirements. Be aware however, that some additional costs will be incurred for the hard convergence exit condition since the sensitivities must be recomputed for the last design cycle.


NX Nastran Input File Processing for Primary and All Auxiliary Models AXMDRV, AXMPR1
IFPL
Does a Design Model Exist?
NO
YES
Generate Nonrepetitive Tables Used in Design Sensitivity and Analysis. Analysis Model Override by Design Model DOPR1, DOPR2, DOPR4, DOM11
PREDOM
1
EXITOPT
Case Control Partitioning Based on Analysis Type MDCASE
PHASE0 Operations, Restart Checking for Primary and Auxiliary Models (for Shape) AXMDRV, AXMPR2
PHASE0
Design Optimization Data Initialization DMPR, SDSA, SDSB
DESINIT, SETSOLAP
Begin Design Cycle Loop
Loop on Design Cycles
Database Cleanup From Previous Design Cycle


Loop on Design Cycle
If AMBS
NO
YES
AMBS ≡ Auxiliary Model Boundary Shapes
Static Analysis for All Auxiliary Models, Loop over Auxiliary Models, Boundary Conditions (FEAOPT = Analysis)
FEA
If AMBS or GMBS
NO
YES
GMBS ≡ Geometry Model Boundary Shapes
Collect Analysis Results, Generate Enforced Boundary Displacements for Auxiliary Structure Analysis AXMDRV, DSAJ
BNDSHP
Finite Element Analysis and Data Recovery for All Analysis Disciplines, Superelements, and Subcases. Basis Vector Solution (for Shape Optimization)
FEA
If AMBS or GMBS
NO
YES
Partition Out Shape Basis Vector Solutions, Generate Basis Vectors
If a Design Model Exists and PARAM, OPTIM, YES
NO
EXIT
YES
Begin Design Sensitivity and Optimization
Pre-Sensitivity Initialization Operations. Shape Basis Vector Initialization and Scaling Operations. DOM11, DOPR2, DOPR3, DOPR4, DOPR5
PRESENS (First Design Cycle Only)


2
Design Constraint Evaluation and Screening
DSAD, DSAR, DSPRM, WEIGHT
DESCON
SDR2STAT
Hard Convergence Check
DOM12
6
(and Design Cycle = 2
e.g., One Pass Through Loop)
EXITOPT
Maximum
Design Cycles
Completed?
YES
EXIT
EXITOPT
7
NO
3
EXITOPT
If Hard
Convergence
YES
Data Recovery and
Exit (FEAOPT =
'DATARECOVERY")
FEA,
EXITOPT
*
7
NO
Generate Pseudo-load Vectors for
Sensitivity Analysis Across Analysis
Disciplines, Subcases, and Superelements
DSABO, DSAE, DSAM, DSAN,
DSAP, DSAR, DSVG1, DSVG2
DESAERDR,
PSLGDV,
SEDRDR
Generate Solutions for
Sensitivity Analysis
(FEAOPT = 'SENSITIVITY')
FEA
Loop on Design Cycle
1


Compute Necessary Response Sensitivities for All Retained Constraints DSAF, DSAH, DSAL, DSAW, DSDVRG, DSFLTF, DSFLUTE, DSPRM, DSVG1, DSVG4,
RESPSEN, SEDRDR, SDR2STAT
4
Exit and Print Sensitivities DSTAP2, DOM6, SDSC
EXITOPT
Optimization with Respect to Approximate Models DOM9
Output Updated Grid Entries DOM11
Design Optimization Output DOM10
Soft Convergence Check DOM12
5
If Soft Convergence
YES
Data Recovery and Exit (FEAOPT = 'DATARECOVERY")
FEA, EXITOPT
7
NO
Analysis Model Updates (Auxiliary + Primary) AXMPR2
UPDATE
End Design Cycle Loop
Loop on Design Cycles

*OPTEXIT=7 includes computation of sensitivities for the current design cycle (all operations up to DOM9).

Chapter 6: Output Features and Interpretation

Since design optimization is an iterative process, a significant amount of output may be generated depending on the quantity of data requested for each design cycle and the number of cycles performed. In design sensitivity and optimization, output may be generated from the following four sources:

1. NX Nastran analysis.

2. Design sensitivity analysis.

3. Design optimization (including optimizer output).

4. Convergence tests and design model updates.

There are a few parameters that affect the level of detail and the frequency of output (for design cycle-dependent data). While covered elsewhere in this Guide, these parameters are collected and summarized in this chapter in Output-Controlling Parameters.

Following this summary, in Design Optimization Output there is an output example from a design optimization problem in the test problem library, which is supplied with your NX Nastran installation software. This example introduces most of the frequently encountered types of design optimization output, within the context of an actual problem. Following in Design Sensitivity Output is a similar output example for design sensitivity analysis.

6.1 Output-Controlling Parameters

Briefly, those quantities that affect either the frequency or level of detail of the output are as follows:

DESPCH	Controls the frequency of DESVAR and GRID Bulk Data entry output to the punch file. The default is for the final design cycle only, but it can be changed to every n-th design cycle (plus the last) with this parameter. See Parameters for Design Sensitivity and Optimization .
IPRINT	Controls the level of printout available directly from the optimizer. Its value, which may range from 0 to 7, is set on the DOPTPRM entry. An increasing value provides increasing levels of detail. See the DOPTPRM entry description in Bulk Data Entries .
NASPRT	Governs the frequency of NX Nastran analysis results output. Output may be requested for the first and last designs only, the first and every n-th design, or turned off completely with this Bulk Data parameter. See Parameters for Design Sensitivity and Optimization .
OPTEXIT	When set to a value of 4 or 7, provides output of the design sensitivity coefficient matrix DSCM2. This Bulk Data parameter may assume any value from 1 to 7 indicating exit at any one of seven predefined locations in the solution sequence. However, only OPTEXIT=4 and 7 generate DSCM2 (design sensitivity coefficient) output. See Parameters for Design Sensitivity and Optimization as well as the Solution 200 flowchart in Solution Sequences .
P1	Controls the frequency of design cycle output. By default, initial and final optimization summaries are output, but it can be changed to every n-th cycle using this parameter. See the DOPTPRM entry in Bulk Data Entries .
P2	Controls how much design cycle information is output at the intervals determined by P1. P2 is set using the DOPTPRM entry. See Bulk Data Entries .

6.2 Design Optimization Output

This section presents design sensitivity and optimization output in the context of one of the example problems from NX Nastran's test problem library. One way to present the optimization output would be to include a listing of all of the various output messages and summaries with a short description of each. However, this approach tends to separate them from their procedural context.

Instead, this section traces selected design optimization output from the stiffened plate example of Stiffened Plate. This problem is chosen because it contains most of the output commonly encountered in a typical optimization example. The best approach is for you to locate this problem in the test problem library and run it on your installation. The output from the actual problem can then be referenced while reading this section.

Note

Your results may differ somewhat depending on your system.

In addition, the Solution 200 flowchart in Solution Sequences might also be useful. Referring to the flowchart will help fix the relationship of the particular output samples to their order in the design optimization process.

D200X7 Output

This example problem is presented in Stiffened Plate. If you have not already done so, you may want to turn to that section first to briefly review the problem set-up.

Assuming you have run this example on your installation and have the output file available, you will notice the following table appearing shortly after the Bulk Data echo:

```
                  -----   COMPARISON BETWEEN INPUT PROPERTY VALUES FROM ANALYSIS AND DESIGN MODELS -----

----------------------------------------------------------------------------------------------------------------
      PROPERTY    PROPERTY      FIELD      ANALYSIS          DESIGN           LOWER           UPPER       DIFFERENCE
        TYPE         ID          ID         VALUE            VALUE            BOUND           BOUND          FLAG
     ------------------------------------------------------------------------------------------------------------------
      PBAR            3           4      1.440000E-01    1.440000E-01    1.000000E-02    1.000000E+20      NONE
      PBAR            3           5      1.728000E-04    1.728000E-04    1.000000E-06    1.000000E+20      NONE
      PBAR            3           6      1.728000E-02    1.728000E-02    1.000000E-06    1.000000E+20      NONE
      PBAR            3           7      1.745000E-02    1.745000E-02    1.000000E-06    1.000000E+20      NONE
      PBAR            3          12      6.000000E-02    6.000000E-02   -1.000000E+35    1.000000E+20      NONE
      PBAR            3          13      6.000000E-01    6.000000E-01   -1.000000E+35    1.000000E+20      NONE
      PBAR            3          14      6.000000E-02    6.000000E-02   -1.000000E+35    1.000000E+20      NONE
      PBAR            3          15     -6.000000E-01   -6.000000E-01   -1.000000E+00    1.000000E+20      NONE
      PBAR            3          16     -6.000000E-02   -6.000000E-02   -1.000000E+00    1.000000E+20      NONE
      PBAR            3          17     -6.000000E-01   -6.000000E-01   -1.000000E+00    1.000000E+20      NONE
      PBAR            3          18     -6.000000E-02   -6.000000E-02   -1.000000E+00    1.000000E+20      NONE
      PBAR            3          19      6.000000E-01    6.000000E-01   -1.000000E+35    1.000000E+20      NONE
      PSHELL          1           4      1.500000E-01    1.500000E-01    1.000000E-02    1.000000E+20      NONE
      PSHELL          2           4      2.000000E-01    2.000000E-01    1.000000E-02    1.000000E+20      NONE
1. IF FIELD ID IS LESS THAN ZERO, IT IDENTIFIES THE WORD POSITION OF AN ENTRY IN EPT.
2. IF FIELD ID IS GREATER THAN ZERO, IT IDENTIFIES THE FIELD POSITION ON A PROPERTY BULK DATA ENTRY.
3. THE DIFFERENCE FLAG IS USED TO CHARACTERIZE DIFFERENCES BETWEEN ANALYSIS AND DESIGN MODEL PROPERTIES:
IF THE FLAG IS NONE, THEN THERE IS NO SIGNIFICANT DIFFERENCE BETWEEN THE TWO VALUES.
IF THE FLAG IS WARNING, THEN THE USER IS ADVISED THAT DIFFERENCES EXIST.
IF THE FLAG IS FATAL, THEN THE DIFFERENCES ARE GREATER THAN  1.00000E+35 AND THE RUN WILL BE TERMINATED.
```

If any of the analysis model properties differs from the design model description, the design model will be used to override the analysis model. This table simply reports on any differences found and provides notification if this override took place. The design value column contains the property values computed from the initial design variable values. In this example, no differences were found.

The override of the analysis model property values by the design model affords a convenient way of performing a type of rudimentary restart. Simply by changing the initial design variable values, you can begin optimization from any point in the design space. (For shape changes, updated GRID entries also need to be included, but these are available from the punch file, if one or more design cycles have been performed in a previous run.) Once the analysis and design models are in agreement, Solution 200 proceeds with an analysis of the model to gather baseline response data. A DMAP information message from subDMAP FEA serves notice that a static analysis has been initiated:

```
^^^ DMAP INFORMATION MESSAGE 9051 (FEA) - STATIC ANALYSIS INITIATED. DESIGN CYCLE NUMBER 1
```

Since this is the first analysis with the parameter NASPRT at its default value of zero, full data recovery is performed based on Case Control output requests. Since displacements and stresses have been requested, we see the familiar NX Nastran static analysis output (abbreviated here):

```
     LOAD CONDITION 1                                                                   SUBCASE 1
                                           D I S P L A C E M E N T   V E C T O R
      POINT ID.   TYPE          T1             T2             T3             R1             R2             R3
         10000      G         .0             .0             .0          -1.927221E-02   2.565913E-03    .0
         10001      G      1.344270E-03  -1.081616E-04    .0           2.904300E-02  -8.063296E-04    .0
         10002      G      2.957638E-03  -3.128633E-04    .0           3.359672E-02   1.434650E-05    .0
         10003      G      5.187130E-03  -1.556333E-04    .0           2.907712E-02   8.084382E-04    .0
         10004      G      8.248678E-03   1.673541E-03    .0          -1.925253E-02  -2.598804E-03    .0
         10100      G         .0         -3.815551E-04    .0           1.309933E-02  -2.005817E-02    .0
          .         .           .              .              .              .              .              .
          .         .           .              .              .              .              .              .
          .         .           .              .              .              .              .              .
          .         .           .              .              .              .              .              .
          .         .           .              .              .              .              .              .
```

```
     LOAD CONDITION 2                                                                   SUBCASE 2
                    S T R E S S E S   I N   Q U A D R I L A T E R A L   E L E M E N T S   ( Q U A D 4 )
       ELEMENT     FIBRE               STRESSES IN ELEMENT COORD SYSTEM             PRINCIPAL STRESSES (ZERO SHEAR)
         ID.      DISTANCE      NORMAL-X       NORMAL-Y      SHEAR-XY      ANGLE         MAJOR           MINOR     VON MISES
          21  -1.000000E-01   3.894159E+03   4.092442E+03   7.646278E+03   45.3714   1.164022E+04   -3.653620E+03  1.383375E+04
               1.000000E-01   3.894159E+03   4.092442E+03   7.646278E+03   45.3714   1.164022E+04   -3.653620E+03  1.383375E+04
          22  -1.000000E-01   1.656648E+02   2.257865E+03   5.013844E+03   50.8926   6.333576E+03   -3.910047E+03  8.953615E+03
               1.000000E-01   1.656648E+02   2.257865E+03   5.013844E+03   50.8926   6.333576E+03   -3.910047E+03  8.953615E+03
          23  -1.000000E-01  -6.033874E+03  -1.339483E+04   7.341767E+03   31.6875  -1.501711E+03   -1.792699E+04  1.722530E+04
               1.000000E-01  -6.033874E+03  -1.339483E+04   7.341767E+03   31.6875  -1.501711E+03   -1.792699E+04  1.722530E+04
```

This is followed by the DMAP information message:

```
^^^ DMAP INFORMATION MESSAGE 9052 (FEA) - STATIC ANALYSIS COMPLETED.DESIGN CYCLE NUMBER= 1
```

If other analyses are performed (normal modes, dynamic response, etc.), similar DMAP information messages and data recovery output will follow.

With the finite element analysis complete, the code could perform a hard convergence check, were this the second or greater design cycle. Since this is just the first cycle though, DOM12 (the module that performs the convergence tests) simply reports on the maximum constraint value:

```
                THIS IS THE FIRST ANALYSIS - NO CONVERGENCE CHECK
                -----------------------------------------------------
                 MAXIMUM VALUE OF CONSTRAINTS           :  1.3119E+00
                -----------------------------------------------------
```

Note

See Eq. 3-6 for a description of normalized constraints used in NX Nastran.

Recall that these constraint values refer to the normalized constraints constructed internally in NX Nastran. Since the maximum constraint value is positive, we can characterize this initial design as infeasible (see the basic optimization problem statement, Getting Started . Furthermore, its value of 1.3119 indicates a constraint violation of around 130%. We will see shortly how to determine which constraint is responsible for this maximum value.

After sensitivity analysis has been performed as indicated by the following DMAP information messages,

```
^^^ DMAP INFORMATION MESSAGE 9051 (FEA) -   STATIC SENSITIVITY ANALYSIS INITIATED.  DESIGN CYCLE NUMBER=            1
^^^ DMAP INFORMATION MESSAGE 9052 (FEA) -   STATIC SENSITIVITY ANALYSIS COMPLETED.  DESIGN CYCLE NUMBER=            1
```

all of the information necessary to perform an optimization is available.

The parameter IPRINT, which can be set using the DOPTPRM entry, controls the amount of diagnostic output from the optimizer. With the default zero value, no output is generated during the numerical optimization phase. In this example, IPRINT has been set to 1 (7 is the highest available value), leading to the following output:

```
 DOT VERSION 4.00

CONTROL PARAMETERS

OPTIMIZATION METHOD,                    METHOD =        1
  NUMBER OF DECISION VARIABLES,              NDV =       3
  NUMBER OF CONSTRAINTS,                    NCON =      55
  PRINT CONTROL PARAMETER,                IPRINT =       1
  GRADIENT PARAMETER,                      IGRAD =       1
    GRADIENTS ARE SUPPLIED BY THE USER
  THE OBJECTIVE FUNCTION WILL BE MINIMIZED
```

Modified method of feasible directions
Set using METHOD on the DOPTPRM entry

Level of optimizer output can be set with IPRINT on the DOPTPRM entry

IGRAD=1 indicates that gradients are supplied to the optimizer. In this case, the “user” is NX Nastran

```
-- INITIAL VARIABLES AND BOUNDS

LOWER BOUNDS ON THE DECISION VARIABLES (XL-VECTOR)
       1)    1.00000E-03   1.00000E-03   1.00000E-03

DECISION VARIABLES (X-VECTOR)
       1)    1.50000E-01   2.00000E-01   1.44000E-01

UPPER BOUNDS ON THE DECISION VARIABLES (XU-VECTOR)
       1)    3.00000E-01   4.00000E-01   2.88000E-01
```

Initial independent design variable data

```
-- INITIAL FUNCTION VALUES

OBJ =   5.7844
```

Initial objective function value

```
CONSTRAINT VALUES (G-VECTOR)
       1)  -4.32761E-01  -4.04487E-01  -4.32761E-01  -4.04487E-01  -4.67287E-01
       6)  -4.54694E-01  -4.67287E-01  -4.54694E-01   1.84507E-01   1.31188E+00
      11)  -3.13682E-01  -4.46650E-01  -3.10988E-01   4.18596E-01  -3.48565E-01
      16)  -9.27064E-02   1.54449E-01  -2.05535E-01   6.86381E-01   7.73185E-01
      21)  -4.46650E-01  -3.10988E-01   4.18596E-01  -3.48565E-01  -9.27064E-02
      26)   1.54449E-01   8.45290E-02  -1.00000E+00  -1.00000E+00  -1.00000E+00
      31)  -1.71800E-01  -1.37363E+00  -1.34228E+00  -3.33333E-01  -3.33333E-01
      36)  -3.33333E-01  -3.33333E-01  -1.00000E+00  -1.00000E+00  -1.00000E+00
      41)  -1.00000E+00  -3.33333E-01  -3.33333E-01  -3.33333E-01  -9.83014E-01
      46)  -3.66569E-01  -3.64299E-01  -1.00000E+00  -1.00000E+00  -1.00000E+00
      51)  -1.00000E+00  -3.33333E-01  -3.33333E-01  -3.33333E-01  -3.33333E-01
```

Initial constraints expressed in normalized form:

$$g(\vec{x}) \leq 0$$

```
-- OPTIMIZATION IS COMPLETE

NUMBER OF ITERATIONS =   21
```

Number of one-dimensional searches performed

```
CONSTRAINT TOLERANCE, CT = -3.00000E-03

THERE ARE       0 ACTIVE CONSTRAINTS AND       5 VIOLATED CONSTRAINTS
  CONSTRAINT NUMBERS
          10        19        20        42        43

THERE ARE       2 ACTIVE SIDE CONSTRAINTS
  VARIABLE NUMBERS (MINUS INDICATES LOWER BOUND)
           1         2
```

Constraint activity at the optimum

```
INATION CRITERIA

21 ITERATIONS FAILED TO PRODUCE A FEASIBLE DESIGN
  OPTIMIZATION TERMINATED

RELATIVE CONVERGENCE CRITERION WAS MET FOR  2 CONSECUTIVE ITERATIONS

ABSOLUTE CONVERGENCE CRITERION WAS MET FOR  2 CONSECUTIVE ITERATIONS
```

ITRMOP on the DOPTPRM entry

```
-- OPTIMIZATION RESULTS

OBJECTIVE, F(X) =    1.12227E+01

DECISION VARIABLES, X

     ID         XL            X           XU
      1     1.00000E-03   3.00000E-01   3.00000E-01
      2     1.00000E-03   4.00000E-01   4.00000E-01
      3     1.00000E-03   1.65920E-01   2.88000E-01

   CONSTRAINTS, G(X)
       1)  -9.40960E-01  -9.19968E-01  -9.40960E-01  -9.19968E-01  -9.51816E-01
       6)  -9.36438E-01  -9.51816E-01  -9.36438E-01  -1.58553E-01   6.27252E-01
      11)  -5.17144E-01  -7.18986E-01  -6.65892E-01  -2.71000E-01  -6.21339E-01
      16)  -4.02165E-01  -4.26015E-01  -4.39304E-01   1.96834E-01   2.59612E-01
      21)  -7.18986E-01  -6.65892E-01  -2.71000E-01  -6.21339E-01  -4.02165E-01
      26)  -4.26015E-01  -3.94653E-01  -3.00000E+00  -3.00000E+00  -1.30444E+00
      31)  -2.28411E-01  -2.15125E+00  -2.10964E+00  -2.84390E-01  -2.84390E-01
      36)  -2.84390E-01  -2.84390E-01  -1.14683E+00  -1.14683E+00  -1.14683E+00
      41)  -1.14683E+00   3.33333E-01   3.33333E-01  -2.31853E-01  -9.77449E-01
      46)  -1.59050E-01  -1.56036E-01  -1.14683E+00  -1.14683E+00  -1.14683E+00
      51)  -1.14683E+00  -2.84390E-01  -2.84390E-01  -2.84390E-01  -2.84390E-01

  FUNCTION CALLS =      49
  GRADIENT CALLS =       3
```

Resultant objective function and constraints based on optimization with respect to the approximate model

This lowest level of optimizer diagnostic output reports on the initial and final design objective, and on the design variable and design constraint values. Since constraints are violated at both the initial and final designs (that is, input to and output from the optimizer, respectively, for this design cycle), it is clear the optimizer was unable to locate a feasible design. Note, however, that the objective function has increased from 5.7844 to 11.2227 in the process. What the optimizer is apparently attempting to do is to minimize the constraint violation by adding weight to the structure. This can be confirmed by observing that all design variables (which are related to sizing features of the structure) have increased in value. However, this may not always be the case. Also, there may be cases where the objective function decreases with improved satisfaction of constraints. All of these are problem-dependent issues.

These conclusions can be confirmed by the summary output for the first design cycle. The frequency and extent of the following output is governed by the parameters P1 and P2. (In this example, P1 = 1, and P2 = 15). This summary begins with a listing of the design objective and the design variables:

```
                   ********************************************************************
                   *                                                                  *
                   *                                                                  *
                   *                 D E S I G N    O P T I M I Z A T I O N           *
                   *                                                                  *
                   *                                                                  *
                    ********************************************************************

                               *******************************************
                              *                                           *
                              *       D E S I G N   C Y C L E       1    *
                              *                                           *
                               *******************************************

                     *****   OPTIMIZATION RESULTS BASED ON THE APPROXIMATE MODEL   *****

                                        -----   DESIGN OBJECTIVE   ------

     -------------------------------------------------------------------------------------------------
         INTERNAL      TYPE                 MINIMIZE
         RESPONSE       OF                     OR       SUPERELEMENT    SUBCASE        INPUT          OUTPUT
            ID       RESPONSE      LABEL     MAXIMIZE        ID            ID          VALUE           VALUE
     --------------------------------------------------------------------------------------------------
            1     DRESP1       WEIGHT    MINIMIZE            0            1      5.7845E+00      1.1223E+01

                                                    -----   DESIGN VARIABLES   ------

     ------------------------------------------------------------------------------------------------------------
            INTERNAL       DESVAR                         LOWER              INPUT            OUTPUT            UPPER
               ID            ID          LABEL            BOUND              VALUE             VALUE            BOUND
     ------------------------------------------------------------------------------------------------------------
                  1             1      T-PLATE        1.0000E-03        1.5000E-01        3.0000E-01        1.0000E+01
                  2             2      T-WEB          1.0000E-03        2.0000E-01        4.0000E-01        1.0000E+01
                  3             3      A-BAR          1.0000E-03        1.4400E-01        1.6592E-01        1.0000E+01
```

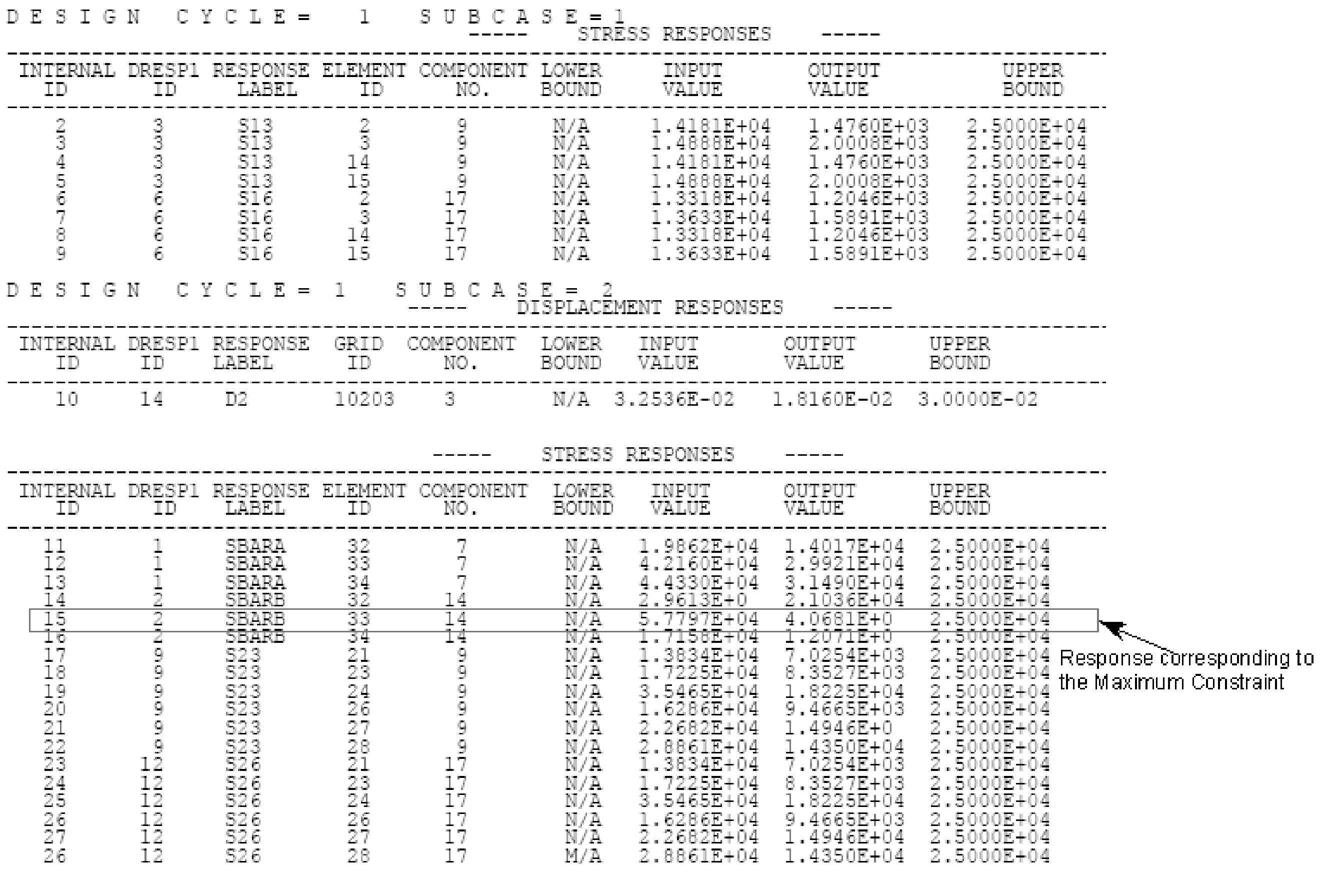
This output supports our observations with respect to the preceding optimizer output, namely, the design objective and design variables have all increased in value on the first design cycle.

Since weight has been added to the structure during the first design cycle, it would be interesting to identify the violated constraints that are driving the design. The following constraints table also appears as part of the summary output and helps us to answer just this sort of question.

```
----   DESIGN CONSTRAINTS ON RESPONSES   -----
                                     (MAXIMUM RESPONSE CONSTRAINTS MARKED WITH **)
----------------------------------------------------------------------------------------------------------------------
                         INTERNAL                           INTERNAL
  INTERNAL     DCONSTR    RESPONSE     RESPONSE     L/U      REGION       SUBCASE        INPUT           OUTPUT
     ID           ID          ID         TYPE       FLAG        ID            ID          VALUE            VALUE
----------------------------------------------------------------------------------------------------------------------
        1           10           2       STRESS     UPPER           2            1      -4.3276E-01      -9.4096E-01
        2           10           3       STRESS     UPPER           2            1      -4.0449E-01      -9.1997E-01
        3           10           4       STRESS     UPPER           2            1      -4.3276E-01      -9.4096E-01
        4           10           5       STRESS     UPPER           2            1      -4.0449E-01      -9.1997E-01
        5           10           6       STRESS     UPPER           2            1      -4.6729E-01      -9.5182E-01
        6           10           7       STRESS     UPPER           2            1      -4.5469E-01      -9.3644E-01
        7           10           8       STRESS     UPPER           2            1      -4.6729E-01      -9.5182E-01
        8           10           9       STRESS     UPPER           2            1      -4.5469E-01      -9.3644E-01
        9           10          14       STRESS     UPPER           1            2       1.8451E-01      -1.5855E-01  maximum
       10           10          15       STRESS     UPPER           1            2       1.3119E+00**     6.2725E-01** normalized
       11           10          16       STRESS     UPPER           1            2      -3.1368E-01      -5.1714E-01  constraint
       12           10          23       STRESS     UPPER           3            2      -4.4665E-01      -7.1899E-01
       13           10          24       STRESS     UPPER           3            2      -3.1099E-01      -6.6589E-01
       14           10          25       STRESS     UPPER           3            2       4.1860E-01      -2.7100E-01
       15           10          26       STRESS     UPPER           3            2      -3.4856E-01      -6.2134E-01
       16           10          27       STRESS     UPPER           3            2      -9.2706E-02      -4.0216E-01
       17           10          28       STRESS     UPPER           3            2       1.5445E-01      -4.2602E-01
       18           10          11       STRESS     UPPER           1            2      -2.0554E-01      -4.3930E-01
       19           10          12       STRESS     UPPER           1            2       6.8638E-01       1.9683E-01
       20           10          13       STRESS     UPPER           1            2       7.7319E-01       2.5961E-01
       21           10          17       STRESS     UPPER           3            2      -4.4665E-01      -7.1899E-01
       22           10          18       STRESS     UPPER           3            2      -3.1099E-01      -6.6589E-01
       23           10          19       STRESS     UPPER           3            2       4.1860E-01      -2.7100E-01
       24           10          20       STRESS     UPPER           3            2      -3.4856E-01      -6.2134E-01
       25           10          21       STRESS     UPPER           3            2      -9.2706E-02      -4.0216E-01
       26           10          22       STRESS     UPPER           3            2       1.5445E-01      -4.2602E-01
       27           30          10         DISP     UPPER          14            2       8.4529E-02      -3.9465E-01

                                   -----   CONSTRAINTS ON DESIGNED PROPERTIES   -----
----------------------------------------------------------------------------------------------------------------------
  INTERNAL    PROPERTY          FIELD           L/U              CYCLE                 INPUT                  OUTPUT
     ID          ID              FLAG            FLAG             LIMIT                 VALUE                  VALUE
----------------------------------------------------------------------------------------------------------------------
       28            1              4          LOWER          7.5000E-02          -1.0000E+00          -3.0000E+00
       29            2              4          LOWER          1.0000E-01          -1.0000E+00          -3.0000E+00
       30            3              4          LOWER          7.2000E-02          -1.0000E+00          -1.3044E+00
       31            3              5          LOWER          1.0000E-06          -1.7180E-01          -2.2841E-01
       32            3              6          LOWER          7.2800E-03          -1.3736E+00          -2.1513E+00
       33            3              7          LOWER          7.4500E-03          -1.3423E+00          -2.1096E+00
       34            3             18          LOWER         -9.0000E-02          -3.3333E-01          -2.8439E-01
       35            3             17          LOWER         -9.0000E-01          -3.3333E-01          -2.8439E-01
       36            3             16          LOWER         -9.0000E-02          -3.3333E-01          -2.8439E-01   move
       37            3             15          LOWER         -9.0000E-01          -3.3333E-01          -2.8439E-01   limits on
       38            3             14          LOWER          3.0000E-02          -1.0000E+00          -1.1468E+00   properties
       39            3             13          LOWER          3.0000E-01          -1.0000E+00          -1.1468E+00
       40            3             12          LOWER          3.0000E-02          -1.0000E+00          -1.1468E+00
       41            3             19          LOWER          3.0000E-01          -1.0000E+00          -1.1468E+00
       42            1              4          UPPER          2.2500E-01          -3.3333E-01           3.3333E-01
       43            2              4          UPPER          3.0000E-01          -3.3333E-01           3.3333E-01
       44            3              4          UPPER          2.1600E-01          -3.3333E-01          -2.3185E-01
       45            3              5          UPPER          1.0173E-02          -9.8301E-01          -9.7745E-01
       46            3              6          UPPER          2.7280E-02          -3.6657E-01          -1.5905E-01
       47            3              7          UPPER          2.7450E-02          -3.6430E-01          -1.5604E-01
       48            3             18          UPPER         -3.0000E-02          -1.0000E+00          -1.1468E+00
       49            3             17          UPPER         -3.0000E-01          -1.0000E+00          -1.1468E+00
       50            3             16          UPPER         -3.0000E-02          -1.0000E+00          -1.1468E+00
       51            3             15          UPPER         -3.0000E-01          -1.0000E+00          -1.1468E+00
       52            3             14          UPPER          9.0000E-02          -3.3333E-01          -2.8439E-01
```

From this summary, note that the maximum constraint (flagged with **) is a result of an upper bound constraint on a stress response. The constraint internal ID is listed as 10, which is simply a means of relating it to the constraint numbering scheme used in the preceding optimizer diagnostic output.

The corresponding internal response ID of this maximum constraint is shown as 15. To see which response this is, the following data is also output at the same time:



```
D E S I G N   C Y C L E =   1   S U B C A S E = 1
                                        -----    STRESS RESPONSES    -----
------------------------------------------------------------------------------------------
 INTERNAL DRESP1 RESPONSE ELEMENT COMPONENT LOWER      INPUT        OUTPUT          UPPER
    ID      ID     LABEL     ID        NO.   BOUND      VALUE        VALUE           BOUND
------------------------------------------------------------------------------------------
    2       3      S13       2        9      N/A    1.4181E+04   1.4760E+03   2.5000E+04
    3       3      S13       3        9      N/A    1.4888E+04   2.0008E+03   2.5000E+04
    4       3      S13      14        9      N/A    1.4181E+04   1.4760E+03   2.5000E+04
    5       3      S13      15        9      N/A    1.4888E+04   2.0008E+03   2.5000E+04
    6       6      S16       2       17      N/A    1.3318E+04   1.2046E+03   2.5000E+04
    7       6      S16       3       17      N/A    1.3633E+04   1.5891E+03   2.5000E+04
    8       6      S16      14       17      N/A    1.3318E+04   1.2046E+03   2.5000E+04
    9       6      S16      15       17      N/A    1.3633E+04   1.5891E+03   2.5000E+04
D E S I G N   C Y C L E = 1   S U B C A S E =  2
                                    -----    DISPLACEMENT RESPONSES    -----
------------------------------------------------------------------------------------------
 INTERNAL DRESP1 RESPONSE   GRID   COMPONENT  LOWER   INPUT        OUTPUT       UPPER
    ID      ID    LABEL      ID       NO.     BOUND   VALUE        VALUE        BOUND
------------------------------------------------------------------------------------------
    10      14     D2       10203     3        N/A  3.2536E-02   1.8160E-02  3.0000E-02


                                      -----    STRESS RESPONSES    -----
------------------------------------------------------------------------------------------
 INTERNAL DRESP1 RESPONSE ELEMENT COMPONENT   LOWER    INPUT        OUTPUT       UPPER
    ID      ID    LABEL      ID       NO.     BOUND    VALUE        VALUE        BOUND
------------------------------------------------------------------------------------------
   11       1     SBARA     32        7        N/A   1.9862E+04   1.4017E+04   2.5000E+04
   12       1     SBARA     33        7        N/A   4.2160E+04   2.9921E+04   2.5000E+04
   13       1     SBARA     34        7        N/A   4.4330E+04   3.1490E+04   2.5000E+04
   14       2     SBARB     32       14        N/A   2.9613E+0    2.1036E+04   2.5000E+04
   15       2     SBARB     33       14        N/A   5.7797E+04   4.0681E+0    2.5000E+04
   16       2     SBARB     34       14        N/A   1.7158E+04   1.2071E+0    2.5000E+04
   17       9     S23       21        9        N/A   1.3834E+04   7.0254E+03   2.5000E+04
   18       9     S23       23        9        N/A   1.7225E+04   8.3527E+03   2.5000E+04
   19       9     S23       24        9        N/A   3.5465E+04   1.8225E+04   2.5000E+04
   20       9     S23       26        9        N/A   1.6286E+04   9.4665E+03   2.5000E+04
   21       9     S23       27        9        N/A   2.2682E+04   1.4946E+0    2.5000E+04
   22       9     S23       28        9        N/A   2.8861E+04   1.4350E+04   2.5000E+04
   23      12     S26       21       17        N/A   1.3834E+04   7.0254E+03   2.5000E+04
   24      12     S26       23       17        N/A   1.7225E+04   8.3527E+03   2.5000E+04
   25      12     S26       24       17        N/A   3.5465E+04   1.8225E+04   2.5000E+04
   26      12     S26       26       17        N/A   1.6286E+04   9.4665E+03   2.5000E+04
   27      12     S26       27       17        N/A   2.2682E+04   1.4946E+04   2.5000E+04
   26      12     S26       28       17        M/A   2.8861E+04   1.4350E+04   2.5000E+04
```

From this output we can see that response number 15 is related to a bar element stress, identified on DRESP1 number 2 for element number 33. Its input value as well as its output are indeed greater than the upper bound of 2.5E4, hence the observed constraint violations. We can conclude that the optimizer has indeed been able to reduce the constraint violation by reducing this peak stress, although at the expense of added weight.

Interpretation of Approximate Optimization Results

It is important to note that the responses in the preceding output are only approximate, because they are based on the approximate model. (See Approximation Concepts in Design Optimization.) The optimizer has been able to achieve a reduction in the peak approximate stresses. These and all other necessary structural responses will be updated again on the next design cycle with a full finite element analysis, and the process will be repeated again.

Once the approximate optimization is complete for this cycle, a soft convergence test is performed, and the results are reported in the following table:

```
                ******************************************************************************
                  INSPECTION OF CONVERGENCE DATA FOR THE OPTIMAL DESIGN WITH RESPECT TO APPROXIMATE MODELS
                                          (SOFT CONVERGENCE DECISION LOGIC)
                      ******************************************************************************

                            RELATIVE CHANGE IN OBJECTIVE    9.4012E-01  MUST BE LESS THAN   1.0000E-03
                       OR   ABSOLUTE CHANGE IN OBJECTIVE    5.4381E+00  MUST BE LESS THAN   1.0000E-02
                      AND   MAX OF RELATIVE PROP.CHANGES    1.0000E+00  MUST BE LESS THAN   1.0000E-03
                                                --- AND ---
                            MAXIMUM CONSTRAINT VALUE        6.2725E-01  MUST BE LESS THAN   5.0000E-03
                       OR   MAX OF RELATIVE D.V. CHANGES    1.0000E+00  MUST BE LESS THAN   1.0000E-03
                      ******************************************************************************
```

Even if soft convergence had been achieved (which it was not), this test will not terminate the design cycles unless the Bulk Data parameter SOFTEXIT had been set to YES. (NO is the default.)

A finite element analysis begins the new design cycle as is apparent from the following DMAP information messages:

```
 ^^^ DMAP INFORMATION MESSAGE 9051 (FEA) -   STATIC ANALYSIS INITIATED.  DESIGN CYCLE NUMBER= 2
0*** USER INFORMATION MESSAGE 5293 FOR DATA BLOCK KLL

LOAD SEQ. NO.           EPSILON             EXTERNAL WORK      EPSILONS LARGER THAN  .001 ARE FLAGGED WITH ASTERISKS
              1          1.0933737E-16          2.1103991E+01
              2         -8.9942808E-16          8.6491135E+01
 ^^^ DMAP INFORMATION MESSAGE 9052 (FEA) -   STATIC ANALYSIS COMPLETED.  DESIGN CYCLE NUMBER=            2
```

With two successive finite element analyses available (one from the current design cycle and one from the previous), a hard convergence test can now be performed. (Recall that this test was skipped on the first design cycle.) Even though convergence has not yet been achieved, the test results are reported nonetheless.

```
 CONVERGENCE NOT ACHIEVED YET   (HARD CONVERGENCE DECISION LOGIC)
             ---------------------------------------------------------------------------------------------

                      RELATIVE CHANGE IN OBJECTIVE          :  9.4027E-01  MUST BE LESS THAN   1.0000E-03
              OR      ABSOLUTE CHANGE IN OBJECTIVE          :  5.4390E+00  MUST BE LESS THAN   1.0000E-02
                                                    --- AND ---
                      MAXIMUM CONSTRAINT VALUE              :  4.0893E-01  MUST BE LESS THAN   5.0000E-03
                                           (CONVERGENCE TO A FEASIBLE DESIGN)
                                                    --- OR ---
                      MAXIMUM OF RELATIVE PROP. CHANGES     :  1.0000E+00  MUST BE LESS THAN   1.0000E-03
              AND     MAXIMUM OF RELATIVE D.V. CHANGES      :  1.0000E+00  MUST BE LESS THAN   1.0000E-03
                                   (CONVERGENCE TO A BEST COMPROMISE INFEASIBLE DESIGN)
             ------------------------------------------------------------------------------------------------
```

In the case of test problem D200X7, the design cycle process is repeated without much incident until cycle number 6 when the following message is issued:

```
********************************************************
*                                                      *
*                 USER  WARNING  MESSAGE               *
*                                                      *
* IF YOU WANT TO RESTART AFTER THIS JOB IS COMPLETED,  *
* YOU MUST INCLUDE A DOPTPRM DATA ENTRY IN THE RESTART *
* BULK DATA WITH THE FOLLOWING ITEMS MODIFIED AS SHOWN:*
*                                                      *
*                  DELP  =   2.5000E-01                *
*                  DPMIN =   5.0000E-03                *
*                  DELX  =   5.0000E-01                *
*                  DXMIN =   2.5000E-02                *
*                                                      *
*  (NOTE: THERE MAY BE MORE THAN ONE MESSAGE LIKE      *
*         THIS.  THE LAST ONE IN THIS RUN SHOULD       *
*         BE LOCATED AND USED.)                        *
********************************************************
```

The above is related to the automatic update of move limits discussed in Optimization with Respect to Approximate Models . Since the design cycle number is greater than two, the constraint violation is increasing, and the maximum constraint violation is greater than 2%, the conclusion is made that the accuracy of the approximate model can be improved if move limits are reduced by 50%. This warning message is just notification that, were optimization to be terminated at this point and then restarted, these new move limits should be included on a new DOPTPRM entry. Note that parameters controlling move limits on properties (DELP, DPMIN) as well as those controlling design variables (DELX, DXMIN) have all been reduced.

One design cycle later, soft convergence is achieved as indicated by the message:

```
                                        ***************************************************
                                      TERMINATION OF DESIGN ITERATION (SOFT CONVERGENCE)
                                      ***************************************************

                          RELATIVE CHANGE IN OBJECTIVE          2.6376E-06 MUST BE LESS THAN   1.0000E-03
                     OR   ABSOLUTE CHANGE IN OBJECTIVE          2.0981E-05 MUST BE LESS THAN   1.0000E-02
                    AND   MAXIMUM OF RELATIVE PROP. CHANGES     0.0000E+00 MUST BE LESS THAN   1.0000E-03
                                                    --- AND ---
                          MAXIMUM CONSTRAINT VALUE              4.1453E-04 MUST BE LESS THAN   5.0000E-03
                     OR   MAXIMUM OF RELATIVE D.V. CHANGES      0.0000E+00 MUST BE LESS THAN   1.0000E-03

               EXPLANATION:
                 THE OPTIMIZATION PROCESS WITH RESPECT TO THE APPROXIMATE MODELS DID NOT CHANGE APPRECIABLY.
                 THIS DESIGN MAY NOT WARRANT AN ADDITIONAL COMPLETE FINITE ELEMENT ANALYSIS.
```

Since SOFTEXIT is NO, this test will not terminate the design cycle process. Rather, another finite element analysis is performed after which hard convergence is indicated:

```
                ***** NORMAL CONVERGENCE CRITERIA SATISFIED ***** (HARD CONVERGENCE DECISION LOGIC)
                 ***************************************************************************************
                                  CONVERGENCE ACHIEVED BASED ON THE FOLLOWING CRITERIA
                                           (HARD CONVERGENCE DECISION LOGIC)

                        RELATIVE CHANGE IN OBJECTIVE        0.0000E+00  MUST BE LESS THAN   1.0000E-03
                  OR    ABSOLUTE CHANGE IN OBJECTIVE        0.0000E+00  MUST BE LESS THAN   1.0000E-02
                                                 --- AND ---
                        MAXIMUM CONSTRAINT VALUE            4.1453E-04  MUST BE LESS THAN   5.0000E-03
                                          (CONVERGENCE TO A FEASIBLE DESIGN)
                                                 --- OR ---
                        MAXIMUM OF RELATIVE PROP. CHANGES   0.0000E+00  MUST BE LESS THAN   1.0000E-03
                  AND   MAXIMUM OF RELATIVE D.V. CHANGES    0.0000E+00  MUST BE LESS THAN   1.0000E-03
                                 (CONVERGENCE TO A BEST COMPROMISE INFEASIBLE DESIGN)
                 ***************************************************************************************
```

Note the similarity between the organization of these convergence messages and the logic in the flowcharts of Relating Design Variables to Properties , Convergence Tests. Since the maximum constraint value is less than .005, we note that we have achieved a feasible optimum solution. And, since the maximum of relative property and design variable changes is less than .001, this design is unique.

After final data recovery and output, which again is the default for a zero value of NASPRT, the following summary of design cycles appears:

```
              ***************************************************************
               S U M M A R Y   O F   D E S I G N   C Y C L E   H I S T O R Y
              ***************************************************************
                                (HARD CONVERGENCE ACHIEVED)
                                (SOFT CONVERGENCE ACHIEVED)
                         NUMBER OF FINITE ELEMENT ANALYSES COMPLETED           8
                         NUMBER OF OPTIMIZATIONS W.R.T. APPROXIMATE MODELS     7
                              OBJECTIVE AND MAXIMUM CONSTRAINT HISTORY
 ---------------------------------------------------------------------------------------------------------
                   OBJECTIVE FROM           OBJECTIVE FROM           FRACTIONAL ERROR           MAXIMUM VALUE
   CYCLE             APPROXIMATE                EXACT                       OF                        OF
   NUMBER           OPTIMIZATION              ANALYSIS                 APPROXIMATION              CONSTRAINT
 ----------------------------------------------------------------------------------------------------------
  INITIAL                                   5.784520E+00                                        1.311878E+00
        1           1.122267E+01            1.122355E+01             -7.868297E-05              4.089278E-01
        2           1.636887E+01            1.636941E+01             -3.285838E-05             -1.610547E-03
        3           1.294620E+01            1.294636E+01             -1.259646E-05             -8.604820E-02
        4           8.748927E+00            8.751337E+00             -2.753791E-04              1.988992E-02
        5           7.379158E+00            7.379156E+00              3.230973E-07              9.885500E-02
        6           7.954716E+00            7.954452E+00              3.315011E-05              4.145312E-04
        7           7.954473E+00            7.954452E+00              2.637622E-06              4.145312E-04
----------------------------------------------------------------------------------------------------------
                                       DESIGN VARIABLE HISTORY
----------------------------------------------------------------------------------------------------------
INTERNAL |  EXTERNAL |          |
 DV. ID. |   DV. ID. |  LABEL   |   INITIAL  : 1     :     2       :    3     :      4     :      5    :
 ---------------------------------------------------------------------------------------------------------
   1     |       1   |  T-PLATE  | 1.5000E-01 : 3.0000E-01: 4.4321E-01 : 2.8106E-01 : 1.4038E-01 :1.0879E-01:
   2     |       2   |  T-WEB    | 2.0000E-01 : 4.0000E-01: 6.1265E-01 : 8.4962E-01 : 8.2809E-01 :7.5169E-01:
   3     |       3   |  A-BAR    | 1.4400E-01 : 1.6592E-01: 1.2682E-01 : 6.4814E-02 : 3.2407E-02 :1.6165E-02:
 -------------------------------------------------------------------------------------------------------
INTERNAL | EXTERNAL |          |
 DV. ID. |  DV. ID. |  LABEL   |    6    :     7    :       8    :      9    :        10    :      11    :
 ---------------------------------------------------------------------------------------------------------
  1      |       1   |  T-PLATE |   1.1169E-01: 1.1169E-01 :
  2      |       2   |  T-WEB   |   8.3887E-01: 8.3887E-01 :
  3      |       3   |  A-BAR   |   1.6165E-02: 1.6165E-02 :
 --------------------------------------------------------------------------------------------------------
```

From this summary we see that an optimal design was found after 7 design cycles and 8 finite element analyses (initial analysis plus one from each design cycle.) As was also seen in the hard convergence output, the final design is feasible.

The final cycle does not always yield the best design. You can identify the best design by examining the objective function values compared with the maximum constraint values. Normally, when the maximum constraint value indicates that a design is feasible, the best design is the one with the minimum (or, depending on the problem's definition, the maximum) objective function. However, if all generated designs are infeasible, then the best design is the one in which there is the lowest maximum constraint violation.

Fractional Error of Approximation: Definition

The objective function history traces the progress made by the optimizer during successive optimizations with respect to approximate models. The fractional error of approximation is a measure of the error in the approximated objective function versus the true objective. The true objective is computed from the analysis at the beginning of the next design cycle. Here, the error is negligible since the weight objective is just a linear function of the design variables. For objectives based on nonlinear structural responses (virtually any response other than weight or volume), the error may be more significant. If these approximation errors are large, tighter move limits may be warranted.

Fractional Error of Approximation: Interpretation

Note also that, as a measure of approximation quality, the fractional error of approximation is incomplete at best. Even though the error in the objective function may be small, errors in constraints may be much greater, especially if the constraints depend on responses that are more nonlinear in the design variables than is the objective. (In fact, this is often the case. For example, weight minimization subject to constraints on stresses has an objective function that is nearly linear in the sizing variables, while the constraints are nonlinear functions of these variables.) Increasing move limits solely on the basis of this information may not bring about the expected increases in efficiency.

The design variable history as a function of design cycle is listed at the end of the summary. The external design variable ID and the label columns are taken directly from the DESVAR Bulk Data entries. The internal design variable ID column simply refers to the order of internal sort on the design variables. (The internal and external design variable sort may differ if DLINK entries are present. Independent design variables are first sorted in ascending numerical order, followed by the dependent design variables, again in ascending order.)

6.3 Design Sensitivity Output

Output for design sensitivity analysis is activated by setting the Bulk Data parameter OPTEXIT to "4 or 7. OPTEXIT = +4 or 7 prints the design sensitivity data to the standard output file and is used primarily for manual interpretation and postprocessing. OPTEXIT = –4 sends the Design sensitivity data to a FORTRAN-readable file using the NX Nastran OUTPUT2 and OUTPUT4 functions. This form is intended for those who would like to couple NX Nastran-computed sensitivities with their own postprocessors, external optimizers, etc.

Design sensitivity output consists of two parts: a matrix of sensitivity coefficients and a table listing the column order of this matrix. The DSCM2 matrix contains these sensitivity coefficients as

$$(DSCM2)_{ij} = \frac{\partial r_j}{\partial x_i}$$

Equation 6-1.

where *i* is the row order and *j* is the column order.

Recall from Design Sensitivity and Optimization in NX Nastran that a sensitivity coefficient is defined as a response rate of change for a corresponding design variable change. (The resultant partial derivative thus gives the slope of the *j*-th response function for the current design in the *i*-th design variable dimension.) The rows of DSCM2 are sorted on ascending independent design variable IDs, given on the DESVAR Bulk Data entries. Row 1 thus corresponds to the independent design variable with the lowest ID, row 2 the design variable with the next higher ID, and so on.

DSCM2 column order is given in a correlation table. For OPTEXIT = +4 or 7, the formatted table contents are printed in the output file. This output indicates the particular column and response correlation in the DSCM2 matrix. For OPTEXIT = –4, this correlation information is output as the DSCMCOL table. See the *NX Nastran Programmer's Manual* for details.

Design Sensitivity Output Example

This example is taken from the Test Problem Library, problem number D200X5. This cantilever plate is presented as an optimization example in Example Problems) and has been modified here by the addition of PARAM,OPTEXIT,4 to the Bulk Data Section. Thus, only sensitivity analysis and not optimization will be performed. See Example Problems for details regarding the problem description and set-up.

D200X5 contains eleven design variables: three independent and eight dependent. The independent design variables are used as multipliers of basis functions, which have been implemented using DLINK entries.

The problem includes weight, displacement, and stress responses, with two separate loading conditions. In short, enough of the basic components are present to give a fairly complete description of the output format in design sensitivity analysis.

The following is an abbreviated listing of the DSCM2 matrix output:

```
MATRIX DSCM2    (GINO NAME 101 ) IS A REAL            61 COLUMN X         3 ROW RECTANG  MATRIX.
0COLUMN      1      ROWS      1 THRU      3     ---------------------------------------------
    ROW
      1)    4.0000E+01  2.2500E+01  1.5938E+01
0COLUMN      2      ROWS      1 THRU      3     ---------------------------------------------
    ROW
      1)    3.2871E+01  1.9345E+01  1.3255E+01
0COLUMN      3      ROWS      1 THRU      3     ---------------------------------------------
    ROW
      1)    3.2871E+01  1.9345E+01  1.3255E+01
0COLUMN      4      ROWS      1 THRU      3     ---------------------------------------------
    ROW
      1)    2.5733E+04  2.9702E+04  3.2985E+04
0COLUMN      5      ROWS      1 THRU      3     ---------------------------------------------
    ROW
      1)    2.5733E+04  2.9702E+04  3.2985E+04
0COLUMN      6      ROWS      1 THRU      3     ---------------------------------------------
    ROW
      1)   -1.5881E+05 -1.5696E+05 -1.5544E+05
0COLUMN      7      ROWS      1 THRU      3     ---------------------------------------------
    ROW
      1)   -1.5881E+05 -1.5696E+05 -1.5544E+05
      .        .           .           .
      .        .           .           .
      .        .           .           .
      .        .           .           .
0COLUMN      39      ROWS      1 THRU      3     ---------------------------------------------
    ROW
      1)   -2.0435E+05 -2.9491E+04 -5.0688E+03
0COLUMN      40      ROWS      1 THRU      3     ---------------------------------------------
    ROW
      1)    2.1453E+00  1.4776E+00  1.1223E+00
0COLUMN      41      ROWS      1 THRU      3     ---------------------------------------------
    ROW
      1)    2.1453E+00  1.4776E+00  1.1223E+00
0COLUMN      42      ROWS      1 THRU      3     ---------------------------------------------
    ROW
      1)   -1.7913E+04 -1.7736E+04 -1.7589E+04
0COLUMN      43      ROWS      1 THRU      3     ---------------------------------------------
    ROW
      1)   -1.7913E+04 -1.7736E+04 -1.7589E+04
0COLUMN      44      ROWS      1 THRU      3     ---------------------------------------------
    ROW
      1)   -1.7913E+04 -1.7736E+04 -1.7589E+04
0COLUMN      45      ROWS      1 THRU      3     ---------------------------------------------
    ROW
      1)   -1.7913E+04 -1.7736E+04 -1.7589E+04
      .        .           .           .
      .        .           .           .
      .        .           .           .
      .        .           .           .
0COLUMN      61      ROWS      1 THRU      3     ---------------------------------------------
    ROW
      1)   -2.1265E+04 -1.0639E+04 -5.4152E+03
0THE NUMBER OF NON-ZERO TERMS IN THE DENSEST COLUMN =        3
0THE DENSITY OF THIS MATRIX IS 100.00 PERCENT.
```

Matrix Dimension

$$\frac{\partial r_3}{\partial x_3} = \left(\frac{\partial u_{z,\,29}}{\partial x_3}\right)_{SUBCASE1}$$

See the following Correlation Table

$$\text{matrix density} = \left(\frac{\text{rows} \cdot \text{columns} - \text{zero terms}}{\text{rows} \cdot \text{columns}}\right) \cdot 100$$

Note that the size of DSCM2 is 3 rows by 61 columns. Each row corresponds to a single independent design variable. Since design variables 9, 10, and 11 are independent, rows 1, 2, and 3 correspond, respectively, to these quantities. Each column corresponds to a particular response; we can note that the sensitivities of 61 responses have been computed. The response order is listed in the correlation table, discussed shortly.

Before moving on to the correlation table, we should note that the density of this matrix is 100%, that is, no zero terms are present. We can conclude that every response will undergo some finite change for a change in any of the independent design variables.

If this density were less than 100%, a closer inspection of the design model formulation might be warranted. A response that is not a function of any of the design variables (hence, a zero sensitivity coefficient), might indicate a design modeling error. Quite often, these types of errors will show up as a null row or column of DSCM2. A null column indicates a response that is not a function of any of the design variables. A null row, on the other hand, indicates that changing a design variable will have no effect whatsoever on any of the design responses. Both of these errors are usually easily addressed and underscore the significance of using design sensitivities as a method for modeling checkout.

Column order in the design sensitivity coefficient matrix is given by the following abbreviated correlation table:

```
-----   IDENTIFICATION OF COLUMNS IN THE DESIGN SENSITIVITY  -----
-----     MATRIX THAT ARE ASSOCIATED WITH DRESP1  ENTRIES    -----
            -----  WEIGHT/VOLUME RESPONSES  -----
         ------------------------------------------
           COLUMN         DRESP1         RESPONSE
            NO.          ENTRY ID          TYPE
         ------------------------------------------
             1             35            WEIGHT
            -----  STATICS RESPONSES  -----
         ------------------------------------------------------------------------------------------------------------
           COLUMN         DRESP1         RESPONSE         GRID/ELM        COMPONENT           SUB             PLY
            NO.          ENTRY ID          TYPE              ID              NO.              CASE             NO.
         ------------------------------------------------------------------------------------------------------------
             2             33              DISP              9                3                1
             3             34              DISP             29                3                1
             4              1            STRESS              1                7                1
             5              1            STRESS             11                7                1
             6              2            STRESS              1                9                1
             7              2            STRESS             11                9                1
             .              .                .               .                .                .
             .              .                .               .                .                .
             .              .                .               .                .                .
             .              .                .               .                .                .
            39             32            STRESS             18               17                1
            40             33              DISP              9                3                2
            41             34              DISP             29                3                2
            42              2            STRESS              1                9                2
            43              2            STRESS             11                9                2
            44              4            STRESS              1               17                2
            45              4            STRESS             11               17                2
             .              .                .               .                .                .
             .              .                .               .                .                .
             .              .                .               .                .                .
             .              .                .               .                .                .
            61             20            STRESS             15               17                2
```

Note that each column is associated with a particular response component for a given subcase. For example, column 3 of DSCM2 contains the sensitivities of the z-component displacement for grid 29, subcase 1. Column 41 contains the sensitivities of the same displacement component for subcase 2.

In general, responses are grouped first by superelement, then by subcase, and finally by DRESP1 or DRESP2 order. A pseudo code of the output loop may be written as follows:

```
for ( superelement = 1, no. of superelements )          {
 for ( load case = 1, no. of load cases )        {
  for ( response = first, last response ) {
   print sensitivity coefficient ;
  }
 }
}
```

The responses are sorted by individual response type. This order is given in Table 1 of the DRESP1 Bulk Data entry description in the *NX Nastran Quick Reference Guide*.

Chapter 7: Example Problems

This chapter includes a wide range of examples intended to highlight and illustrate the Design Sensitivity and Optimization features of the software. Though by no means exhaustive, these eleven examples cover most major features including both Property and Shape Optimization, Dynamic Response Optimization, Superelements, Composites, Acoustic Responses, and Restarts. Aeroelastic Optimization examples are presented in the *NX Nastran Aeroelastic Analysis User's Guide*.

The results in this chapter were obtained on a workstation which may be different than your workstation. Although your results may differ slightly, running these examples on your own machine is a good way to familiarize yourself with some of the software's capabilities. All of these decks are available in the NX Nastran Test Problem Library, delivered with your system. If you are unable to find these files, you should contact your System Administrator for assistance.

Another good way to learn is by modifying these decks. You might try adding Shape variables to a Property Optimization example, adding analysis disciplines, modifying the constraints, and so on. Most of these decks are relatively small, and should not incur any great CPU-related costs. In any event, the time spent learning with simple examples can more than pay for itself when faced with larger, more complex, problems.

Examples in this chapter include:

Three-Bar Truss	TPL problem D200X1.DAT This is a classic getting started example. It examines weight minimization of a structure subject to stress and displacement constraints from multiple statics subcases. Design Variables are linearly related to rod cross-sectional areas. It introduces explicit design variable linking.
Vibration of a Cantilever Beam (Turner's Problem)	TPL problem D200X6.DAT Examines weight minimization of a structure subject to a free-vibration frequency constraint. Design variables are linearly related to rod areas and shell thickness. Discusses interpretation of weight objective function if concentrated masses are used.
Cantilevered Plate	TPL problem D200X5.DAT Illustrates reduced-basis formulation using explicit design variable linking (DLINK). Weight minimization of a statically loaded plate with constraints on principal and von Mises stresses as well as grid displacements.
Stiffened Plate	TPL problem D200X7.DAT Illustrates effective grouping of elements by property type for design modeling purposes and use of the DVPREL2 entry for user-defined property relations. Weight minimization subject to constraints on static stresses and displacements. Design variables are related to plate thicknesses and bar cross-sectional properties.
Shape Optimization of a Culvert	TPL problems D200CSX.DAT, D200CS.DAT Illustrates Shape Optimization using the Direct Input of Shapes method. The objective is weight minimization subject to von Mises stress constraints.

Analytic Boundary Shapes	TPL problem D200AM3.DAT Illustrates Shape Optimization using the Analytic Boundary Shapes method to minimize the weight of a solid cantilever subject to constraints on von Mises stresses.
Dynamic Response Optimization	TPL problem D200RML3.DAT Uses synthetic responses to express a minimum RMS objective in Modal Frequency Analysis.
Twenty-Five Bar Truss, Superelement Optimization	TPL problem D200X3S.DAT Based on the 25 Bar Truss, TPL problem D200X3, this example illustrates superelement optimization of a structure subject to constraints on local Euler buckling. Also illustrates the use of synthetic responses and constraint scaling.
Design Optimization with Composite Materials	TPL problem D200C01.DAT Examines modifying ply thicknesses and orientations to achieve an optimal design based on Hill failure-type constraints.
Acoustic Optimization	TPL problem ACCOPT1.DAT Presents a method for minimization of acoustic pressure response peaks over a range of frequencies. This approach is also quite useful for other dynamic response applications.
Restarts in Design Optimization	Based on TPL problem D200X1.DAT Illustrates basic restart capabilities available in Solution 200. Discusses pseudo-restart from a new location in the Design Space and restarts from an OPTEXIT point.

7.1 Three-Bar Truss

A common task in design optimization is to reduce the mass of a structure subjected to not just one, but several load conditions. Figure 7-1 shows a simple three-bar truss that must be built to withstand two separate loading conditions. Note that these two loads subject the outer truss members to both compressive as well as tensile loads. Due to the loading symmetry, we expect the design to be symmetric as well. As an exercise, we'll show how to enforce this symmetry using design variable linking.

An important, but often overlooked consideration is that the optimization capability in NX Nastran is multidisciplinary. That is, the final optimal design is the result of a simultaneous consideration of all analysis disciplines across all subcases. In this case, the optimal three-bar truss design will satisfy the load requirements for both statics subcases, which is to be expected. (If, for example, a normal modes subcase were to be added, the resultant design would have to not only satisfy the static strength requirements, but also constraints on eigenvalues. As an exercise you may wish to try adding an eigenvalue constraint.

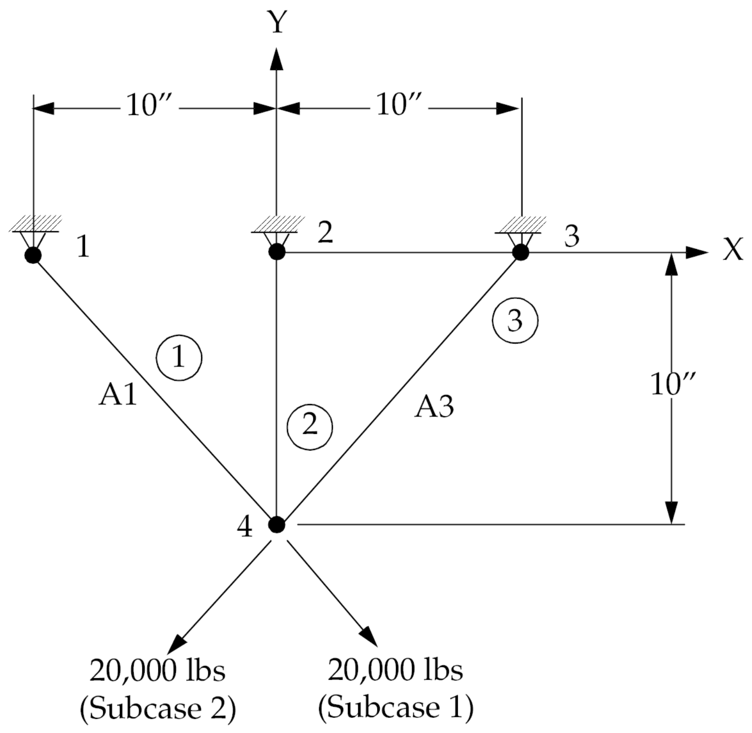


Figure 7-1. Three-Bar Truss

Analysis Model Description

Three rod (pin-jointed) structure in the x-y plane
Symmetric structural configuration with respect to the y-axis
Two distinct 20,000 lb load conditions

Material:	E = 1.0E7 psi
	Weight density = 0.1 lbs/in^3

Design Model Description

Objective:	Minimization of structural weight	
Design variable:	Cross-sectional areas A_1, A_2, and A_3	
	Design variable linking such that $A_3 = A_1$	
Constraints:	Allowable stress:	Tensile 20,000 psi
		Compressive = -15,000 psi

The input data for this problem is given in Listing 7-1. Grid, element, and load data are assigned based on the data supplied in the figure. The two separate static load cases are defined in Solution 200 Case Control using the parameter ANALYSIS = STATICS.

Turning to the design model description, we see three design variables are used, one each to control the individual rod element cross-sectional areas. The set of DESVAR, DVPREL1 entries define the relations:

$$A_i = 1.0X_i \qquad i = 1, 2, 3$$

Equation 7-1.

Note that the initial design variable values are equal to the initial rod cross-sectional areas. With the multiplying coefficients of 1.0 on the DVPREL1 entries, the design model and the initial analysis model properties agree. Thus, no design model override takes place.

Since the model's geometry and loading are both symmetric, we expect the optimizer to yield a symmetric final design with A1 = A3. In a more complex structure, we may want to enforce some type of continuity ourselves, perhaps to address ease-of-manufacture concerns. This can be done using the DLINK entry to explicitly link design variables together. Although not strictly necessary, DLINK is used to define the relation

$$X_3 = 1.0X_1$$

Equation 7-2.

The value of X_3 now depends on X_1. Therefore, the optimizer only needs to consider two independent design variables X_1 and X_2 instead of three. This not only simplifies the problem, but also reduces the cost of the sensitivity analysis. This cost savings can be significant in a larger problem.

An alternate method of linking would have been placing rod elements 1and 3 in the same property group. The DLINK entry would then be unnecessary since a single design variable could control the areas of all elements of the group. You may want to try this formulation yourself to see that the results are indeed identical.

DRESP1 entries define the design responses: the displacement and stress responses that are to be constrained and the weight response that is to be used as the objective. Note that the objective and constraints are called out in Case Control. The DESOBJ command points to the weight response defined on DRESP1 20, while the DESSUB command identifies DCONSTR set 21. Since this appears above the subcase level, the constraint bounds will be applied to both subcases.

A DOPTPRM entry is used to override some of the default optimization parameters. DESMAX, or the maximum number of design cycles to be performed, has been increased from the default of 5 to 10. (If convergence is indicated, fewer than 10 cycles may be performed.) DELP has also been increased from its default of 0.2 to 0.5. This allows any analysis model parameter to undergo changes of up to 50% on each design cycle. This provides move limits on the approximate model. And finally, IPRINT, P1, and P2 have all been increased from their defaults to allow more diagnostic output.

Turning to the results of the optimization, we see from the hard convergence decision logic output that the problem converges to a feasible design since all constraints are satisfied at the optimum. However, note that the properties and design variables have still changed appreciably from their values on the previous design cycle. This suggests that we may have some flexibility in choosing final dimensions for our design without greatly affecting the overall weight.

The summary of design cycle history and the design variable history tables both indicate that convergence was achieved in five iterations. A total of six finite element analyses were performed, one for the initial analysis and one in connection with each of the design cycles. Also listed in the summary of design cycle history is the fractional error of approximation in the objective function, which compares the approximate objective function on exit from the optimizer with the true value computed from the subsequent finite element analysis. This is an indication of the quality of the objective function approximation. Since the weight is linear in the design variables ($W = L_1X_1 + L_2X_2 + L_3L_3$), we expect the linearized weight objective to be an exact approximation; this indeed is seen to be the case, except for a small amount of numerical "noise." Caution should be used when interpreting the data appearing in the fractional error of approximation column.

One might be tempted to use the fractional error of approximation information as justification for increasing the move limits (DELP and DELX on the DOPTPRM entry, and/or DELXV on the DESVAR entry) in order to achieve convergence earlier. This might not yield the expected results. In this case, the linear objective function was approximated linearly, which is exact. However, the stress constraints in this problem are proportional to the inverse of the cross-sectional areas, and are thus nonlinear in the design variables. Too large of a move limit could lead to increased approximation errors as was illustrated in Chapter 3. Here, the mixed method of approximation was used, which we know to be pretty good at approximating sizing-type stress constraints. Thus, increasing move limits to 50% is reasonable, as it will probably save us the cost of a few additional finite element analyses. A recommendation would be to stick with the defaults, unless something is known about the problem beforehand that would indicate enhanced efficiency by changing them.

```
         ***** NORMAL CONVERGENCE CRITERIA SATISFIED ***** (HARD CONVERGENCE DECISION LOGIC)
      **************************************************************************************
                       CONVERGENCE ACHIEVED BASED ON THE FOLLOWING CRITERIA
                                (HARD CONVERGENCE DECISION LOGIC)
             RELATIVE CHANGE IN OBJECTIVE        2.7611E-03  MUST BE LESS THAN   1.0000E-03
       OR    ABSOLUTE CHANGE IN OBJECTIVE        7.4799E-03  MUST BE LESS THAN   1.0000E-02
                                        --- AND ---
             MAXIMUM CONSTRAINT VALUE           -9.2676E-05  MUST BE LESS THAN   5.0000E-03
                               (CONVERGENCE TO A FEASIBLE DESIGN)
                                        --- OR ---
             MAXIMUM OF RELATIVE PROP. CHANGES   1.5165E-01  MUST BE LESS THAN   1.0000E-03
       AND   MAXIMUM OF RELATIVE D.V. CHANGES    1.5165E-01  MUST BE LESS THAN   1.0000E-03
                      (CONVERGENCE TO A BEST COMPROMISE INFEASIBLE DESIGN)
      **************************************************************************************
                        ***************************************************************
                        S U M M A R Y   O F   D E S I G N   C Y C L E   H I S T O R Y
                        ***************************************************************
                                           (HARD CONVERGENCE ACHIEVED)
                            NUMBER OF FINITE ELEMENT ANALYSES COMPLETED           6
                            NUMBER OF OPTIMIZATIONS W.R.T. APPROXIMATE MODELS     5
                                   OBJECTIVE AND MAXIMUM CONSTRAINT HISTORY
---------------------------------------------------------------------------------------------------------------
               OBJECTIVE FROM      OBJECTIVE FROM      FRACTIONAL ERROR      MAXIMUM VALUE         ROW OF
  CYCLE          APPROXIMATE           EXACT                 OF                   OF               MAXIMUM
  NUMBER        OPTIMIZATION          ANALYSIS          APPROXIMATION         CONSTRAINT          CONSTRAINT
---------------------------------------------------------------------------------------------------------------
  INITIAL                          4.828427E+00                             -3.234952E-01            2
        1        3.007858E+00      3.008487E+00         -2.093749E-04       -3.737109E-03            2
        2        2.820099E+00      2.819910E+00          6.696224E-05       -1.902266E-02            3
        3        2.735039E+00      2.735147E+00         -3.957449E-05       -7.482129E-03            3
        4        2.709055E+00      2.709030E+00          9.152918E-06       -2.097656E-04            3
        5        2.701577E+00      2.701550E+00          9.884281E-06       -9.267578E-05            2
---------------------------------------------------------------------------------------------------------------
```

```
ID UGS, D200X1
TIME  10      $
SOL 200       $  OPTIMIZATION
CEND
TITLE = SYMMETRIC THREE BAR TRUSS DESIGN OPTIMIZATION  -  D200X1
SUBTITLE = BASELINE - 2 CROSS SECTIONAL AREAS AS DESIGN VARIABLES
ECHO   = SORT
SPC    = 100
DISP   = ALL
STRESS = ALL
DESOBJ(MIN) = 20 $  (DESIGN OBJECTIVE = DRESP ID)
DESSUB      = 21 $  DEFINE CONSTRAINT SET FOR BOTH SUBCASES
ANALYSIS = STATICS
SUBCASE 1
   LABEL = LOAD CONDITION 1
   LOAD  = 300
SUBCASE 2
   LABEL = LOAD CONDITION 2
   LOAD  = 310
BEGIN BULK
$
$------------------------------------------------------------------------
$ ANALYSIS MODEL
$------------------------------------------------------------------------
$
$ GRID DATA
$       2       3       4       5       6       7       8       9         10
GRID,   1,      ,       -10.0 ,   0.0,  0.0
GRID,   2,      ,         0.0 ,   0.0,  0.0
GRID,   3,      ,        10.0 ,   0.0,  0.0
GRID,   4,      ,         0.0 , -10.0,  0.0
$ SUPPORT DATA
SPC,    100,    1,      123456, ,       2,      123456
SPC,    100,    3,      123456, ,       4,      3456
$ ELEMENT DATA
CROD,   1,      11,     1,      4
```

```
CROD,   2,      12,     2,      4
CROD,   3,      13,     3,      4
$ PROPERTY DATA
PROD,   11,     1,      1.0
PROD,   12,     1,      2.0
PROD,   13,     1,      1.0
MAT1,   1,      1.0E+7, ,       0.33,   0.1
$ EXTERNAL LOADS DATA
FORCE,  300,    4,      ,       20000.,  0.8,   -0.6
FORCE,  310,    4,      ,       20000., -0.8,   -0.6
$
$-----------------------------------------------------------------------
$ DESIGN MODEL
$-----------------------------------------------------------------------
$
$...DESIGN VARIABLE DEFINITION
$DESVAR,ID,     LABEL,  XINIT,  XLB,    XUB,    DELXV(OPTIONAL)
DESVAR, 1,      A1,     1.0,    0.1,    100.0
DESVAR, 2,      A2,     2.0,    0.1,    100.0
DESVAR, 3,      A3,     1.0,    0.1,    100.0
$
$...IMPOSE X3=X1 (LEADS TO A3=A1)
$DLINK, ID,     DDVID,  CO,     CMULT,  IDV1,   C1,     IDV2,   C2,    +
$+,     IDV3,   C3,     ...
DLINK,  1,      3,      0.0,    1.0,    1,      1.00
$
Listing 7-1.  (Cont.)
$...DEFINITION OF DESIGN VARIABLE TO ANALYSIS MODEL PARAMETER RELATIONS
$DVPREL1,ID,    TYPE,   PID,    FID,    PMIN,   PMAX,   C0,     ,      +
$+,     DVID1,  COEF1,  DVID2,  COEF2,  ...
DVPREL1,10,     PROD,   11,     4,      ,       ,       ,       ,   +DP1
+DP1,   1,      1.0
DVPREL1,20,     PROD,   12,     4,      ,       ,       ,       ,         +DP2
+DP2,   2,      1.0
DVPREL1,30,     PROD,   13,     4,      ,       ,       ,       ,         +DP3
+DP3,   3,      1.0
$
$...STRUCTURAL RESPONSE IDENTIFICATION
$DRESP1,ID,     LABEL,  RTYPE,  PTYPE,  REGION, ATTA,   ATTB,   ATT1,  +
$+,     ATT2,   ...
DRESP1, 20,     W ,     WEIGHT
DRESP1, 21,     U4,     DISP  , ,       ,       1,      ,       4
DRESP1, 22,     V4,     DISP  , ,       ,       2,      ,       4
DRESP1, 23,     S1,     STRESS, PROD,   ,       2,      ,       11
DRESP1, 24,     S2,     STRESS, PROD,   ,       2,      ,       12
DRESP1, 25,     S3,     STRESS, PROD,   ,       2,      ,       13
$
$...CONSTRAINTS
$DCONSTR,DCID,  RID,    LALLOW, UALLOW
DCONSTR,21,     21,     -0.20  ,0.20
DCONSTR,21,     22,     -0.20  ,0.20
DCONSTR,21,     23,     -15000.,20000.
DCONSTR,21,     24,     -15000.,20000.
DCONSTR,21,     25,     -15000.,20000.
$
$...OPTIMIZATION CONTROL:
$
DOPTPRM,IPRINT, 5,      DESMAX, 10,     DELP,   0.5,    P1,     1,     +
+,      P2,     15
$
$.......2.......3.......4.......5.......6.......7.......8.......9......0
ENDDATA
```

Listing 7-1.

7.2 Vibration of a Cantilever Beam (Turner's Problem)

This problem was originally published by M.J. Turner ("Design of Minimum Mass Structures with Specified Natural Frequencies," *AIAA Journal* Vol. 5, No. 3, March 1967). The problem is to design a minimum weight structure while constraining the fundamental bending frequency at or above 20 Hz. The beam is symmetric and made up of a shear web having top and bottom caps that are modeled with rod elements. Turner's original design model consisted of piecewise linear bar cross-sectional areas and web thicknesses; however, we will just approximate this as a step function model with uniform cross-sectional rod elements and uniform thickness shear elements within each of three bays.

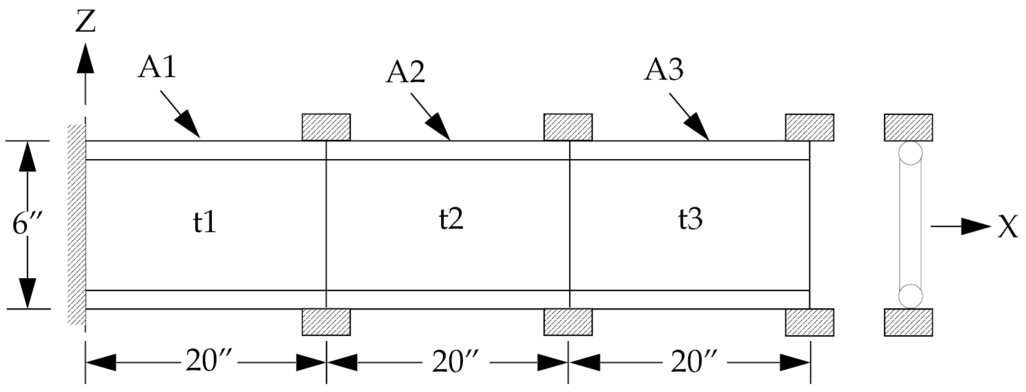


Figure 7-2. Cantilever Beam Vibration Model.

Analysis Model Description

Web is modeled by CQUAD4 elements
Caps are modeled by ROD elements

Material:	E = 1.03E7 psi
	Poisson ratio = 0.3
	Weight density = 0.1 lbs/in^3

Non-structural mass: Lumped masses at top and bottom nodes, 15 lbs. each

Design Model Description

Objective:	Minimization of structural weight
Design variables:	Cross-sectional areas of bar elements (symmetry imposed on the top and bottom elements)
Constraints:	Fundamental bending frequency at or above 20 Hz

Normal modes analysis is requested in Solution 200 by setting the Case Control parameter ANALYSIS = MODES. In order to determine a minimum weight design subject to a lower-bound constraint on fundamental frequency, the thicknesses of the three shear panels and the six rod cross-sectional areas are allowed to vary. These variations are described using six design variables; three for the web thicknesses and three for the rod cross-sectional areas (symmetry is imposed on the upper and lower ROD element areas.) See Listing 7-2 for details.

The lower-bound constraint on the first mode is f=20 Hz. Note that the DCONSTR entry places bounds on $\lambda = (2\pi f)^2$, or 15,791 (radians/sec^2). The upper bound will not drive the design, so it is just set to an arbitrarily large value of 2.0E5.

It should be noted that the weight (or, more appropriately, mass) objective function is computed in NX Nastran based on a (volume • density) calculation. Thus, elements that have no volume are not included in the weight response calculation. Here, the mass contribution of the concentrated masses is ignored, as can be verified from the summary of design cycle history. In general, although concentrated and distributed masses are not included in the design model responses, they are accounted for, as expected, in the analysis.

Turning to the optimization results, we note that hard convergence has been achieved after five design cycles, requiring a total of six finite element analyses. We note that a feasible design has been achieved as the maximum constraint value is less than the allowable maximum. Again, the final "weight" objective is that due to the plate and rod elements only, excluding the concentrated masses.

```
                  ***** NORMAL CONVERGENCE CRITERIA SATISFIED ***** (HARD CONVERGENCE DECISION LOGIC)
                 **************************************************************************************
                                  CONVERGENCE ACHIEVED BASED ON THE FOLLOWING CRITERIA
                                           (HARD CONVERGENCE DECISION LOGIC)
                        RELATIVE CHANGE IN OBJECTIVE        2.0467E-04  MUST BE LESS THAN   1.0000E-03
                  OR    ABSOLUTE CHANGE IN OBJECTIVE        1.4267E-03  MUST BE LESS THAN   1.0000E-02
                                                     --- AND ---
                        MAXIMUM CONSTRAINT VALUE            1.7216E-03  MUST BE LESS THAN   5.0000E-03
                                          (CONVERGENCE TO A FEASIBLE DESIGN)
                                                     --- OR ---
                        MAXIMUM OF RELATIVE PROP. CHANGES   2.3337E-02  MUST BE LESS THAN   1.0000E-03
                  AND   MAXIMUM OF RELATIVE D.V. CHANGES    2.3337E-02  MUST BE LESS THAN   1.0000E-03
                                 (CONVERGENCE TO A BEST COMPROMISE INFEASIBLE DESIGN)
                 **************************************************************************************

                                   ***************************************************************
                                   S U M M A R Y   O F   D E S I G N   C Y C L E   H I S T O R Y
                                   ***************************************************************
                                                       (HARD CONVERGENCE ACHIEVED)
                                       NUMBER OF FINITE ELEMENT ANALYSES COMPLETED           6
                                       NUMBER OF OPTIMIZATIONS W.R.T. APPROXIMATE MODELS     5
                                              OBJECTIVE AND MAXIMUM CONSTRAINT HISTORY
           ---------------------------------------------------------------------------------------------------------------
                          OBJECTIVE FROM       OBJECTIVE FROM       FRACTIONAL ERROR      MAXIMUM VALUE         ROW OF
             CYCLE          APPROXIMATE            EXACT                  OF                   OF               MAXIMUM
             NUMBER        OPTIMIZATION          ANALYSIS            APPROXIMATION         CONSTRAINT          CONSTRAINT
           ---------------------------------------------------------------------------------------------------------------
             INITIAL                             1.920000E+01                               0.000000E+00            0
                   1        9.599560E+00         9.599990E+00       -4.480287E-05          9.583312E-02            1
                   2        7.749448E+00         7.749540E+00       -1.187549E-05          3.574277E-03            1
                   3        7.054639E+00         7.054621E+00        2.568502E-06         -5.288688E-03            1
                   4        6.970634E+00         6.970623E+00        1.436540E-06          1.697157E-03            1
                   5        6.969195E+00         6.969197E+00       -2.736827E-07          1.721585E-03            1

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


DESIGN VARIABLE HISTORY
-----------------------------------------------------------------------------------------------------------------------------
  INTERNAL |   EXTERNAL   |              |
   DV. ID. |    DV. ID.   |    LABEL     |   INITIAL    :      1       :      2       :     3       :      4       :      5     :
-----------------------------------------------------------------------------------------------------------------------------
         1 |         1    |  A1          |   1.0000E+00 :   5.0000E-01 :   7.5000E-01 :  8.0722E-01 :   8.2632E-01 : 8.2653E-01 :
         2 |         2    |  A2          |   1.0000E+00 :   5.0000E-01 :   3.5193E-01 :  4.4035E-01 :   4.4570E-01 : 4.4566E-01 :
         3 |         3    |  A3          |   1.0000E+00 :   5.0000E-01 :   2.5000E-01 :  1.2494E-01 :   1.1281E-01 : 1.1269E-01 :
         4 |         4    |  T1          |   2.0000E-01 :   1.0000E-01 :   9.5153E-02 :  5.8634E-02 :   5.1032E-02 : 5.0650E-02 :
         5 |         5    |  T2          |   2.0000E-01 :   1.0000E-01 :   5.0000E-02 :  4.6748E-02 :   4.3546E-02 : 4.3211E-02 :
         6 |         6    |  T3          |   2.0000E-01 :   1.0000E-01 :   5.0000E-02 :  2.5000E-02 :   2.4701E-02 : 2.5277E-02 :
-----------------------------------------------------------------------------------------------------------------------------
*** USER INFORMATION MESSAGE 6464 (DOM12E)
    RUN TERMINATED DUE TO HARD CONVERGENCE TO A POSSIBLY NON-UNIQUE OPTIMUM AT CYCLE NUMBER =        5.
```

```
nastran oldq4k
ID UGS, D200X6
TIME  10
SOL 200       $  OPTIMIZATION
CEND
TITLE    = VIBRATION OF A BEAM.
ECHO     = UNSORT
DISP     = ALL
STRESS   = ALL
METHOD   = 1
ANALYSIS = MODES
DESOBJ(MIN) = 1    $ OBJECTIVE FUNCTION DEFINITION
DESSUB =      10   $ CONSTRAINT SET SELECTION
$
BEGIN BULK
$
$------------------------------------------------------------------------
$ ANALYSIS MODEL:
$------------------------------------------------------------------------
$
EIGRL,  1,      ,       ,
GRID,   1,      ,       0.0,    0.0,    -3.0,   ,       123456
GRID,   2,      ,       20.0,   0.0,    -3.0,   ,       2456
GRID,   3,      ,       40.0,   0.0,    -3.0,   ,       2456
GRID,   4,      ,       60.0,   0.0,    -3.0,   ,       2456
GRID,   5,      ,       0.0,    0.0,    3.0,    ,       123456
```

```
GRID,   6,      ,       20.0,   0.0,    3.0,    ,       2456
GRID,   7,      ,       40.0,   0.0,    3.0,    ,       2456
GRID,   8,      ,       60.0,   0.0,    3.0,    ,       2456
$
CROD,   1,      201,    5,      6
CROD,   2,      202,    6,      7
CROD,   3,      203,    7,      8
CROD,   7,      201,    1,      2
CROD,   8,      202,    2,      3
CROD,   9,      203,    3,      4
PROD,   201,    1,      1.0,    0.0
PROD,   202,    1,      1.0,    0.0
PROD,   203,    1,      1.0,    0.0
$
CQUAD4, 4,      204,    1,      2,      6,      5
CQUAD4, 5,      205,    2,      3,      7,      6
CQUAD4, 6,      206,    3,      4,      8,      7
PSHELL, 204,    1,      0.2
PSHELL, 205,    1,      0.2
PSHELL, 206,    1,      0.2
$
CONM2,  10,     2,      ,       15.0
CONM2,  11,     3,      ,       15.0
CONM2,  12,     4,      ,       15.0
CONM2,  14,     6,      ,       15.0
CONM2,  15,     7,      ,       15.0
CONM2,  16,     8,      ,       15.0
$
MAT1,   1,      1.03E7, ,       0.3,    0.1
PARAM,  WTMASS, 0.002588
PARAM,  GRDPNT, 1
$
$-----------------------------------------------------------------------
$ DESIGN MODEL:
$-----------------------------------------------------------------------
$
$...Define the design variables
$DESVAR,ID,     LABEL,  XINIT,  XLB,    XUB,    DELXV
$
DESVAR, 1,      A1,     1.0,    0.01,   100.0
DESVAR, 2,      A2,     1.0,    0.01,   100.0
DESVAR, 3,      A3,     1.0,    0.01,   100.0
DESVAR, 4,      T1,     0.2,    0.001,  10.0
DESVAR, 5,      T2,     0.2,    0.001,  10.0
DESVAR, 6,      T3,     0.2,    0.001,  10.0
$
$...Relate the design variables to analysis model properties
$DVPREL1,ID,    TYPE,   PID,    FID,    PMIN,   PMAX,   C0,     ,       +
$+,     DVIDD1, COEF1,  DVID2,  COEF2,  ...
$
DVPREL1,1,      PROD,   201,    4,      ,       ,       ,       ,   +DP1
+DP1,   1,      1.0
DVPREL1,2,      PROD,   202,    4,      ,       ,       ,       ,   +DP2
+DP2,   2,      1.0
DVPREL1,3,      PROD,   203,    4,      ,       ,       ,       ,   +DP3
+DP3,   3,      1.0
DVPREL1,4,      PSHELL, 204,    4,      ,       ,       ,       ,   +DP4
+DP4,   4,      1.0
DVPREL1,5,      PSHELL, 205,    4,      ,       ,       ,       ,   +DP5
+DP5,   5,      1.0
DVPREL1,6,      PSHELL, 206,    4,      ,       ,       ,       ,   +DP6
+DP6,   6,      1.0
$
$...Identify the analysis responses to be used in the design model
$DRESP1,ID,     LABEL,  RTYPE,  PTYPE,  REGION, ATTA,   ATTB,   ATT1,  +
$+,     ATT2,   ...
$
DRESP1, 1,      W,      WEIGHT
DRESP1, 2,      F1,     EIGN,    ,      ,       1
$
$...Use these responses to define the objective (in case control)
$   and the constraints:
$DCONSTR,DCID,  RID,    LALLOW, UALLOW
DCONSTR,10,     2,      15791., 200000.0    $ lower bound = 20 Hz
$
$...Optional override of design optimization parameters:
DOPTPRM, IPRINT,2,      DESMAX, 10,     DELP,   0.5,    P1,     1,     +
+,      P2,    15
$
$.......2.......3.......4.......5.......6.......7.......8.......9......0
ENDDATA
```

Listing 7-2.

7.3 Cantilevered Plate

Reduced basis formulations were introduced in Approximation Concepts in Design Optimization as an efficient way to reduce the number of independent design variables. This example shows how DLINK entries can be used to express these reduced basis formulations.

Suppose we wish to determine the optimum thickness distribution of the cantilever plate in Figure 7-3 such that the structural mass is minimized. Two separate loading conditions exist: the first is a tip loading condition as shown in the figure; the second is a uniform pressure loading in the -Z direction. Constraints are placed on maximum allowable tip displacement and von Mises stresses at the upper and lower surfaces of each shell element.

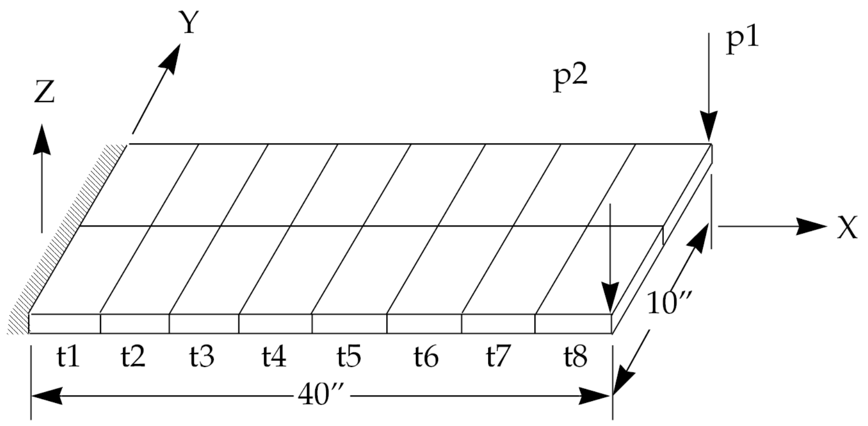


Figure 7-3. Cantilever Plate Thickness Design

Analysis Model Description

2 x 8 array of CQUAD4 elements

Material:	E = 1.0E+6 psi	
	Poisson ratio = 0.33	
	Weight density = 0.1 lbs/in^3	
Two static load conditions:		
	Tip loading:	Two 5,000 lb loads in -z direction
	Pressure loading:	Uniform at -6.44 lb/in^2

Design Model Description

Objective:	Structural weight minimization	
Design variables:	Basis functions:	Constant, linear and quadratic in x direction to describe plate element thicknesses Explicit design variable linking via DLINK entries
Constraints:	Major principal and von Mises stresses at lower-plate surfaces (Z1)	
	Minor principal and von Mises stresses at upper-plate surfaces (Z2)	

The reduced basis functions used here consist of three basis vectors: constant, linear, and quadratic. The thickness distribution can be written as a linear combination of these basis vectors as:

$$\begin{Bmatrix} t_1 \\ t_2 \\ t_3 \\ \vdots \\ t_8 \end{Bmatrix} = \alpha_1 \begin{Bmatrix} 1.0 \\ 1.0 \\ 1.0 \\ \vdots \\ 1.0 \end{Bmatrix} + \alpha_2 \begin{Bmatrix} 1.0 \\ 0.875 \\ 0.75 \\ \vdots \\ 0.125 \end{Bmatrix} + \alpha_3 \begin{Bmatrix} 1.0 \\ 0.7656 \\ 0.5625 \\ \vdots \\ 0.0156 \end{Bmatrix}$$

Equation 7-3.

The eight individual plate thicknesses (one for each station along the length of the cantilever) are functions of three independent design variables α_1 , α_2 and α_3 . One way of implementing Eq. 7-3 is with DLINK entries as shown in Listing 7-1. Another way of writing this reduced basis formulation is with DVPREL1 entries alone (you may want to try this exercise on your own). However, by using DLINK entries and a dependent design variable for each plate thickness, the final values of the dependent variables correspond exactly to the final plate thickness distribution. This is helpful in evaluating the results, as is shown in the design variable history summary.

The dependent design variables, defined using DESVAR entries 1 through 8, are assigned initial values that correspond to the initial thickness distribution (see the PSHELL entries 1 through 8). This has been done for clarity only. Actually, the initial thickness distribution is only a function of the independent variables ALPHA1, ALPHA2, and ALPHA3 and the constants defined on the DLINK entries. If differences were to exist, the dependent variables would be overridden accordingly, in addition to the corresponding plate element thicknesses.

The summary of design cycle history indicates that the initial design is infeasible. After seven iterations, a minimum weight design has been achieved that satisfies all constraints. However, note that in order to achieve an optimal, feasible design, the optimizer still found it necessary to add mass to the structure. The result is a minimum weight structure that still satisfies all of the constraints.

```
                  ***** NORMAL CONVERGENCE CRITERIA SATISFIED ***** (HARD CONVERGENCE DECISION LOGIC)
                  *************************************************************************************
                                   CONVERGENCE ACHIEVED BASED ON THE FOLLOWING CRITERIA
                                            (HARD CONVERGENCE DECISION LOGIC)
                         RELATIVE CHANGE IN OBJECTIVE        4.1124E-04  MUST BE LESS THAN   1.0000E-03
                   OR    ABSOLUTE CHANGE IN OBJECTIVE        1.0109E-01  MUST BE LESS THAN   1.0000E-01
                                                   --- AND ---
                         MAXIMUM CONSTRAINT VALUE            2.4815E-03  MUST BE LESS THAN   5.0000E-03
                                           (CONVERGENCE TO A FEASIBLE DESIGN)
                                                   --- OR ---
                         MAXIMUM OF RELATIVE PROP. CHANGES   5.2540E-04  MUST BE LESS THAN   1.0000E-03
                   AND   MAXIMUM OF RELATIVE D.V. CHANGES    5.2540E-04  MUST BE LESS THAN   1.0000E-03
                                  (CONVERGENCE TO A BEST COMPROMISE INFEASIBLE DESIGN)
                  *************************************************************************************
```

```
                                   ***************************************************************
                                   S U M M A R Y   O F   D E S I G N   C Y C L E   H I S T O R Y
                                   ***************************************************************
                                                   (HARD CONVERGENCE ACHIEVED)
                                                   (SOFT CONVERGENCE ACHIEVED)
                                       NUMBER OF FINITE ELEMENT ANALYSES COMPLETED           9
                                       NUMBER OF OPTIMIZATIONS W.R.T. APPROXIMATE MODELS     8
                                              OBJECTIVE AND MAXIMUM CONSTRAINT HISTORY
           ---------------------------------------------------------------------------------------------------------------
                              OBJECTIVE FROM           OBJECTIVE FROM          FRACTIONAL ERROR          MAXIMUM VALUE
             CYCLE              APPROXIMATE                EXACT                      OF                      OF
             NUMBER            OPTIMIZATION               ANALYSIS               APPROXIMATION            CONSTRAINT
           ---------------------------------------------------------------------------------------------------------------
             INITIAL                                    7.843750E+01                                      1.072155E+01
                   1            1.134805E+02            1.134805E+02             4.033855E-07             4.562702E+00
                   2            1.575682E+02            1.575683E+02            -6.778743E-07             1.796172E+00
                   3            2.029126E+02            2.029123E+02             1.503979E-06             5.925344E-01
                   4            2.365482E+02            2.365477E+02             2.193210E-06             1.399380E-01
                   5            2.438008E+02            2.438006E+02             8.136332E-07             2.652916E-02
                   6            2.435833E+02            2.435836E+02            -1.190216E-06             3.164406E-02
                   7            2.458150E+02            2.458149E+02             1.862229E-07             3.082301E-03
                   8            2.459160E+02            2.459160E+02            -6.204878E-08             2.481479E-03
           ---------------------------------------------------------------------------------------------------------------
```

```
                                                         DESIGN VARIABLE HISTORY
 ---------------------------------------------------------------------------------------------------------------------------------
  INTERNAL |   EXTERNAL   |              |
   DV. ID. |    DV. ID.   |    LABEL     |   INITIAL    :      1       :      2       :     3       :      4       :      5     :
 ---------------------------------------------------------------------------------------------------------------------------------
         1 |          9   |  ALPHA1      |   1.0000E+00 :   1.4186E+00 :   1.7602E+00 :  1.8056E+00 :   2.0352E+00 : 1.3748E+00 :
         2 |         10   |  ALPHA2      |   1.0000E+00 :   1.5778E+00 :   3.1178E+00 :  6.1321E+00 :   7.8664E+00 : 1.1305E+01 :
         3 |         11   |  ALPHA3      |   1.0000E+00 :   1.3324E+00 :   1.0673E+00 : -4.5704E-01 :  -1.3711E+00 :-4.1134E+00 :
         4 |          1   |  T1          |   3.0000E+00 :   4.3288E+00 :   5.9453E+00 :  7.4806E+00 :   8.5304E+00 : 8.5666E+00 :
         5 |          2   |  T2          |   2.6406E+00 :   3.8193E+00 :   5.3054E+00 :  6.8213E+00 :   7.8685E+00 : 8.1175E+00 :
         6 |          3   |  T3          |   2.3125E+00 :   3.3514E+00 :   4.6989E+00 :  6.1476E+00 :   7.1637E+00 : 7.5399E+00 :
         7 |          4   |  T4          |   2.0156E+00 :   2.9252E+00 :   4.1257E+00 :  5.4596E+00 :   6.4161E+00 : 6.8337E+00 :
         8 |          5   |  T5          |   1.7500E+00 :   2.5406E+00 :   3.5859E+00 :  4.7574E+00 :   5.6256E+00 : 5.9990E+00 :
         9 |          6   |  T6          |   1.5156E+00 :   2.1977E+00 :   3.0795E+00 :  4.0409E+00 :   4.7922E+00 : 5.0358E+00 :
        10 |          7   |  T7          |   1.3125E+00 :   1.8964E+00 :   2.6064E+00 :  3.3101E+00 :   3.9161E+00 : 3.9440E+00 :
        11 |          8   |  T8          |   1.1406E+00 :   1.6367E+00 :   2.1666E+00 :  2.5650E+00 :   2.9970E+00 : 2.7237E+00 :
 ---------------------------------------------------------------------------------------------------------------------------------
  INTERNAL |   EXTERNAL   |              |
   DV. ID. |    DV. ID.   |    LABEL     |        6     :         7    :         8    :        9    :        10    :        11 :
 ---------------------------------------------------------------------------------------------------------------------------------
         1 |          9   |  ALPHA1      |   7.7168E-01 :   7.5696E-01 :   7.5705E-01 :
         2 |         10   |  ALPHA2      |   1.3916E+01 :   1.3964E+01 :   1.3968E+01 :
         3 |         11   |  ALPHA3      |  -6.2996E+00 :  -6.1894E+00 :  -6.1892E+00 :
         4 |          1   |  T1          |   8.3884E+00 :   8.5311E+00 :   8.5356E+00 :
         5 |          2   |  T2          |   8.1253E+00 :   8.2363E+00 :   8.2402E+00 :
         6 |          3   |  T3          |   7.6654E+00 :   7.7481E+00 :   7.7514E+00 :
         7 |          4   |  T4          |   7.0086E+00 :   7.0664E+00 :   7.0692E+00 :
         8 |          5   |  T5          |   6.1549E+00 :   6.1914E+00 :   6.1936E+00 :
         9 |          6   |  T6          |   5.1044E+00 :   5.1229E+00 :   5.1246E+00 :
        10 |          7   |  T7          |   3.8570E+00 :   3.8610E+00 :   3.8622E+00 :
        11 |          8   |  T8          |   2.4128E+00 :   2.4057E+00 :   2.4063E+00 :
 ---------------------------------------------------------------------------------------------------------------------------------
 *** USER INFORMATION MESSAGE 6464 (DOM12E)
     RUN TERMINATED DUE TO HARD CONVERGENCE TO AN OPTIMUM AT CYCLE NUMBER =        8.
```

```
ID UGS, D200X5
TIME  10
SOL 200       $  OPTIMIZATION
CEND
TITLE    = CANTILEVERED PLATE  -  D200X5
SUBTITLE = REDUCED BASIS FORMULATION
SPC      = 100
OLOAD    = ALL
DISP     = ALL
STRESS   = ALL
DESOBJ(MIN,MAX) = 35   $ OBJECTIVE FUNCTION DEFINITION
SUBCASE 1
   ANALYSIS = STATICS
   LABEL    = LOAD CONDITION 1
   LOAD     = 300
   DESSUB   = 10
SUBCASE 2
   ANALYSIS = STATICS
   LABEL    = LOAD CONDITION 2
   LOAD     = 310
   DESSUB   = 10
BEGIN BULK
$
$-----------------------------------------------------------------------
$ ANALYSIS MODEL:
$-----------------------------------------------------------------------
$
MAT1,   51,     1.0E+7, ,       0.33,   0.1,    ,       ,       ,       +M2
+M2,    50000., 50000., 29000.
SPC1,   100,    123456, 1,      11,     21
GRID,   1 ,     ,        0.,    -5.,    0.
```

```
GRID,   2 ,     ,        5.,    -5.,    0.
GRID,   3 ,     ,       10.,    -5.,    0.
GRID,   4 ,     ,       15.,    -5.,    0.
GRID,   5 ,     ,       20.,    -5.,    0.
GRID,   6 ,     ,       25.,    -5.,    0.
GRID,   7 ,     ,       30.,    -5.,    0.
GRID,   8 ,     ,       35.,    -5.,    0.
GRID,   9 ,     ,       40.,    -5.,    0.
GRID,   11,     ,        0.,     0.,    0.
GRID,   12,     ,        5.,     0.,    0.
GRID,   13,     ,       10.,     0.,    0.
GRID,   14,     ,       15.,     0.,    0.
GRID,   15,     ,       20.,     0.,    0.
GRID,   16,     ,       25.,     0.,    0.
GRID,   17,     ,       30.,     0.,    0.
GRID,   18,     ,       35.,     0.,    0.
GRID,   19,     ,       40.,     0.,    0.
GRID,   21,     ,        0.,     5.,    0.
GRID,   22,     ,        5.,     5.,    0.
GRID,   23,     ,       10.,     5.,    0.
GRID,   24,     ,       15.,     5.,    0.
GRID,   25,     ,       20.,     5.,    0.
GRID,   26,     ,       25.,     5.,    0.
GRID,   27,     ,       30.,     5.,    0.
GRID,   28,     ,       35.,     5.,    0.
GRID,   29,     ,       40.,     5.,    0.
$
CQUAD4,  1,     1,       1,      2,     12,     11
CQUAD4,  2,     2,       2,      3,     13,     12
CQUAD4,  3,     3,       3,      4,     14,     13
CQUAD4,  4,     4,       4,      5,     15,     14
CQUAD4,  5,     5,       5,      6,     16,     15
CQUAD4,  6,     6,       6,      7,     17,     16
CQUAD4,  7,     7,       7,      8,     18,     17
CQUAD4,  8,     8,       8,      9,     19,     18
CQUAD4, 11,     1,      11,     12,     22,     21
CQUAD4, 12,     2,      12,     13,     23,     22
CQUAD4, 13,     3,      13,     14,     24,     23
CQUAD4, 14,     4,      14,     15,     25,     24
CQUAD4, 15,     5,      15,     16,     26,     25
CQUAD4, 16,     6,      16,     17,     27,     26
CQUAD4, 17,     7,      17,     18,     28,     27
CQUAD4, 18,     8,      18,     19,     29,     28
PSHELL,  1,     51,     3.0,     51,      ,     51
PSHELL,  2,     51,     2.640625,51
PSHELL,  3,     51,     2.3125,  51
PSHELL,  4,     51,     2.015625,51
PSHELL,  5,     51,     1.75,    51
PSHELL,  6,     51,     1.515625,51
PSHELL,  7,     51,     1.3125,  51
PSHELL,  8,     51,     1.140625,51
$       2       3       4       5       6       7       8       9         10
FORCE,  300,    9,      ,       50000., 0.0,    0.0,    -1.0
FORCE,  300,    29,     ,       50000., 0.0,    0.0,    -1.0
PLOAD2, 310,    -60.44,  1,     THRU,    8
PLOAD2, 310,    -60.44, 11,     THRU,   18
$
$----------------------------------------------------------------------
$ DESIGN MODEL:
$----------------------------------------------------------------------
$
$...Define the design variables
$DESVAR,ID,     LABEL,  XINIT,  XLB,    XUB,    DELXV
$ (This group will be the dependent design variables:)
DESVAR,  1,     T1,     3.0,     0.001, 100.0
DESVAR,  2,     T2,     2.640625,0.001, 100.0
DESVAR,  3,     T3,     2.3125,  0.001, 100.0
DESVAR,  4,     T4,     2.015625,0.001, 100.0
DESVAR,  5,     T5,     1.75,    0.001, 100.0
DESVAR,  6,     T6,     1.515625,0.001, 100.0
DESVAR,  7,     T7,     1.3125,  0.001, 100.0
DESVAR,  8,     T8,     1.140625,0.001, 100.0
$ (This group will be the independent design variables:)
DESVAR,  9,     ALPHA1, 1.0,    -1.+10, 1.+10
DESVAR, 10,     ALPHA2, 1.0,    -1.+10, 1.+10
DESVAR, 11,     ALPHA3, 1.0,    -1.+10, 1.+10
$
$...Explicit design variable linking
$
$DLINK, ID,     DDVID,  CO,     CMULT,  IDV1,   C1,     IDV2,   C2,     +
$+,     IDV3,   C3,     ...
DLINK,  1 ,     1,      ,       ,       9,      1.0,    10,     1.0,    +DL1
+DL1 ,  11,     1.0
DLINK,  2 ,     2,      ,       ,       9,      1.0,    10,     0.8750, +DL2
+DL2 ,  11,     0.765625
DLINK,  3 ,     3,      ,       ,       9,      1.0,    10,     0.7500, +DL3
+DL3 ,  11,     0.5625
DLINK,  4 ,     4,      ,       ,       9,      1.0,    10,     0.6250, +DL4
+DL4 ,  11,     0.390625
DLINK,  5 ,     5,      ,       ,       9,      1.0,    10,     0.5000, +DL5
+DL5 ,  11,     0.2500
DLINK,  6 ,     6,      ,       ,       9,      1.0,    10,     0.3750, +DL6
```

```
+DL6 ,  11,     0.140625
DLINK,  7 ,     7,      ,       ,       9,      1.0,    10,     0.2500, +DL7
+DL7 ,  11,     0.0625
DLINK,  8 ,     8,      ,       ,       9,      1.0,    10,     0.1250, +DL8
+DL8 ,  11,     0.015625
$
$...Express analysis model properties linearly in terms of design variables
$DVPREL1,ID,    TYPE,   PID,    FID,    PMIN,   PMAX,   C0,     ,       +
$+,     DVID1,  COEF1,  DVID2,  COEF2,  ...
DVPREL1,1,      PSHELL, 1,      4,      ,       ,       ,       ,       +DP1
+DP1,   1,      1.0
Listing 7-3.  (Cont.)
DVPREL1,2,      PSHELL, 2,      4,      ,       ,       ,       ,       +DP2
+DP2,   2,      1.0
DVPREL1,3,      PSHELL, 3,      4,      ,       ,       ,       ,       +DP3
+DP3,   3,      1.0
DVPREL1,4,      PSHELL, 4,      4,      ,       ,       ,       ,       +DP4
+DP4,   4,      1.0
DVPREL1,5,      PSHELL, 5,      4,      ,       ,       ,       ,       +DP5
+DP5,   5,      1.0
DVPREL1,6,      PSHELL, 6,      4,      ,       ,       ,       ,       +DP6
+DP6,   6,      1.0
DVPREL1,7,      PSHELL, 7,      4,      ,       ,       ,       ,       +DP7
+DP7,   7,      1.0
DVPREL1,8,      PSHELL, 8,      4,      ,       ,       ,       ,       +DP8
+DP8,   8,      1.0
$
$...Identify the design responses
$DRESP1,ID,     LABEL,  RTYPE,  PTYPE,  REGION, ATTA,   ATTB,   ATT1,   +
$+,     ATT2,   ...
DRESP1,  1,     S11,    STRESS, PSHELL, ,        7,     ,       1
DRESP1,  2,     S12,    STRESS, PSHELL, ,        9,     ,       1
DRESP1,  3,     S13,    STRESS, PSHELL, ,       16,     ,       1
DRESP1,  4,     S14,    STRESS, PSHELL, ,       17,     ,       1
DRESP1,  5,     S21,    STRESS, PSHELL, ,        7,     ,       2
DRESP1,  6,     S22,    STRESS, PSHELL, ,        9,     ,       2
DRESP1,  7,     S23,    STRESS, PSHELL, ,       16,     ,       2
DRESP1,  8,     S24,    STRESS, PSHELL, ,       17,     ,       2
DRESP1,  9,     S31,    STRESS, PSHELL, ,        7,     ,       3
DRESP1, 10,     S32,    STRESS, PSHELL, ,        9,     ,       3
DRESP1, 11,     S33,    STRESS, PSHELL, ,       16,     ,       3
DRESP1, 12,     S34,    STRESS, PSHELL, ,       17,     ,       3
DRESP1, 13,     S41,    STRESS, PSHELL, ,        7,     ,       4
DRESP1, 14,     S42,    STRESS, PSHELL, ,        9,     ,       4
DRESP1, 15,     S43,    STRESS, PSHELL, ,       16,     ,       4
DRESP1, 16,     S44,    STRESS, PSHELL, ,       17,     ,       4
DRESP1, 17,     S51,    STRESS, PSHELL, ,        7,     ,       5
DRESP1, 18,     S52,    STRESS, PSHELL, ,        9,     ,       5
DRESP1, 19,     S53,    STRESS, PSHELL, ,       16,     ,       5
DRESP1, 20,     S54,    STRESS, PSHELL, ,       17,     ,       5
DRESP1, 21,     S61,    STRESS, PSHELL, ,        7,     ,       6
DRESP1, 22,     S62,    STRESS, PSHELL, ,        9,     ,       6
DRESP1, 23,     S63,    STRESS, PSHELL, ,       16,     ,       6
DRESP1, 24,     S64,    STRESS, PSHELL, ,       17,     ,       6
DRESP1, 25,     S71,    STRESS, PSHELL, ,        7,     ,       7
DRESP1, 26,     S72,    STRESS, PSHELL, ,        9,     ,       7
DRESP1, 27,     S73,    STRESS, PSHELL, ,       16,     ,       7
DRESP1, 28,     S74,    STRESS, PSHELL, ,       17,     ,       7
DRESP1, 29,     S81,    STRESS, PSHELL, ,        7,     ,       8
DRESP1, 30,     S82,    STRESS, PSHELL, ,        9,     ,       8
DRESP1, 31,     S83,    STRESS, PSHELL, ,       16,     ,       8
DRESP1, 32,     S84,    STRESS, PSHELL, ,       17,     ,       8
DRESP1, 33,     D1 ,    DISP  , ,       ,        3,     ,       9
DRESP1, 34,     D2 ,    DISP  , ,       ,        3,     ,       29
$
$...Define the response to be used as the objective function:
DRESP1, 35,     W  ,    WEIGHT
$
$...Define the design constraints
$DCONSTR,DCID,  RID,    LALLOW, UALLOW
DCONSTR,10,      1,     -50000.,50000.
DCONSTR,10,      2,     -29000.,29000.
DCONSTR,10,      3,     -50000.,50000.
DCONSTR,10,      4,     -29000.,29000.
DCONSTR,10,      5,     -50000.,50000.
DCONSTR,10,      6,     -29000.,29000.
DCONSTR,10,      7,     -50000.,50000.
DCONSTR,10,      8,     -29000.,29000.
DCONSTR,10,      9,     -50000.,50000.
DCONSTR,10,     10,     -29000.,29000.
DCONSTR,10,     11,     -50000.,50000.
DCONSTR,10,     12,     -29000.,29000.
DCONSTR,10,     13,     -50000.,50000.
DCONSTR,10,     14,     -29000.,29000.
DCONSTR,10,     15,     -50000.,50000.
DCONSTR,10,     16,     -29000.,29000.
DCONSTR,10,     17,     -50000.,50000.
DCONSTR,10,     18,     -29000.,29000.
DCONSTR,10,     19,     -50000.,50000.
DCONSTR,10,     20,     -29000.,29000.
DCONSTR,10,     21,     -50000.,50000.
DCONSTR,10,     22,     -29000.,29000.
```

```
DCONSTR,10,     23,     -50000.,50000.
DCONSTR,10,     24,     -29000.,29000.
DCONSTR,10,     25,     -50000.,50000.
DCONSTR,10,     26,     -29000.,29000.
DCONSTR,10,     27,     -50000.,50000.
DCONSTR,10,     28,     -29000.,29000.
DCONSTR,10,     29,     -50000.,50000.
DCONSTR,10,     30,     -29000.,29000.
DCONSTR,10,     31,     -50000.,50000.
DCONSTR,10,     32,     -29000.,29000.
DCONSTR,10,     33,     -2.,    2.
DCONSTR,10,     34,     -2.,    2.
$
$...Override optimization parameter defaults (optional)
DOPTPRM,DESMAX, 10,     DELP,   0.5,    DPMIN,  .01,    DELX,   2.0,    +
+,      DELB,   0.01,   CONV2,  0.1
$
ENDDATA
```

Listing 7-3.

7.4 Stiffened Plate

As discussed in Design Modeling for Sensitivity and Optimization, an effective way to keep the number of independent design variables to a minimum is by grouping designed elements by property type. A smaller set of independent design variables decreases the cost associated with the sensitivity analysis, allows the optimizer to perform more efficiently, and makes interpretation of the final results much easier.

A simple example is shown in Figure 7-4. The design goal is to reduce the weight of the stiffened panel subject to stress and displacement constraints under two separate static load conditions. The thickness of the plate and web, and the cross-sectional area of the web cap are all allowed to vary. The boundary conditions are selected so as to model an infinite plate. The first load case includes both uniaxial tension in the x-direction and a vertical pressure load in the z-direction. The second load case is a concentrated load applied in the +z direction at grid 10203, which is directly under the web.

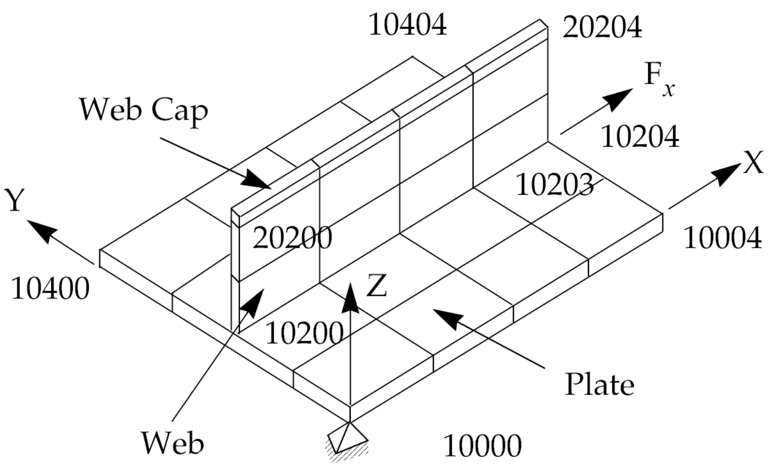


Figure 7-4. Stiffened Plate Model

Analysis Model Description

16 CQUAD4 element to model uniform thickness base plate

8 CQUAD4 elements to model the uniform thickness web
4 CROD elements to model the uniform cross-sectional area web cap

Material:	E = 1.0E7 psi	
	Poisson ratio = 0.33	
	Weight density = 0.283 lbs/in^3	
Loadings:	Subcase 1:	In-plane tensile load of Nx = 1000 lbs/in
		Uniform pressure load of 50 psi over the plate in the positive z-direction
	Subcase 2:	Lumped vertical load of 10,000 lbs at Grid 10203

Design Model Description

Objective:	Structural weight minimization
Design variables:	Plate thickness, web thickness, and web cap cross-sectional properties
Constraints:	
Strength:	von Mises stress ≤ 25,000 psi at the centroid of plate elements for Subcase 1
	Maximum axial stress ≤ 25,000 psi at both ends of rod elements for Subcase 2
Displacement:	Vertical displacement at grid point 10302 for Subcase 1 ≤ 0.1 inches
	Vertical displacement at grid point 10203 for Subcase 2 ≤ 0.03 inches

The DVPREL1 entries 1, 2, and 3 each express an analysis model property directly in terms of a given design variable:

$$\begin{aligned} t_{plate} &= 1.0X_1 \\ t_{web} &= 1.0X_2 \\ A_{cap} &= 1.0X_3 \end{aligned}$$

Equation 7-4.

If the cap section is assumed to have a rectangular cross-section, then the other cross-sectional properties can be expressed directly in terms of the area Design Variable. Figure 7-5 shows a rectangular cross section characterized by the dimensions b and h, with four stress recovery points C, D, E, and F.

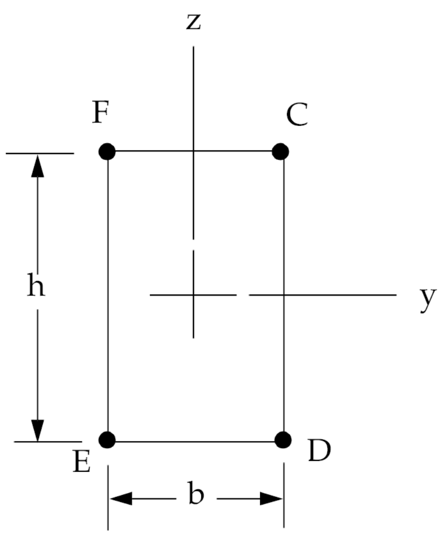


Figure 7-5.

The initial area can be written in terms of the original dimensions as

$$A^0 = b^0 \cdot h^0$$

Equation 7-5.

For a proportional change Δ in the cross-sectional dimensions *b* and *h* , the corresponding area is

$$\begin{aligned} A &= \Delta b^0 \cdot \Delta h^0 \\ &= \Delta^2 b^0 h^0 \end{aligned}$$

Equation 7-6.

or

$$\Delta = \left(\frac{A}{A^0}\right)^{1/2}$$

Equation 7-7.

To look at it another way, for a given change in the design variable, area, we immediately know how the underlying cross-sectional dimensions change. This can be used to express the changes in area moments of inertia in terms of Δ, or:

$$\begin{aligned} I_1 &= \frac{(\Delta b^0)^3(\Delta h^0)}{12} \qquad (=I_{zz}) \\ &= \Delta^4 I_1^0 \end{aligned}$$

Equation 7-8.

where:

$$\Delta = \left(\frac{A}{A^0}\right)^{1/2} = \left(\frac{X_3}{X_3^0}\right)^{1/2}$$

Equation 7-9.

and so on.

Since bending stresses in the web cap are to be used in the design model as constraints, the stress recovery points must move as the area changes. If not, then we fail to recover stresses at the outer corners of the cross section where bending stresses are greatest. For point C we have, in terms of Δ:

$$\begin{aligned} C_y &= \Delta C_y^0 \\ C_z &= \Delta C_z^0 \end{aligned}$$

Equation 7-10.

and so on for the other stress recovery points, D, E, and F.

Since the above relations are nonlinear in the design variable X_3, we need to express these using DVPREL2 Bulk Data entries. These entries provide the input arguments to the DEQATN entries on which the preceding relations have been defined. In addition, the DTABLE Bulk Data entry has been used to define constants that appear in the equations. We could have provided these constants directly on the DEQATN Bulk Data entries instead, but the loss of generality would have required us to use additional equations.

In order for a structural response to be used either as an objective or a constraint, it first must be identified on a DRESPi Bulk Data entry. The DRESP1 entries 1 and 2, for example, identify the maximum stress at ends A and B of a bar element. (Plot codes are used in the ATTA fields of the DRESP1 entry. For the BAR element, Item 7 is the maximum stress at end A, and Item14 is the maximum stress at end B. See Appendix A of the *NX Nastran Quick Reference Guide* for a list of these plot codes.) The ATTi fields on these entries identify the property ID; here it is 3. DRESP1 number 1 indicates that the maximum end A stress for every BAR element in PBAR group 3 is to be used in the design model for a total of four such responses. DRESP1 number 2 identifies similar information for end B of the BAR elements.

Similarly, DRESP1 entries 3, 6, 9, and 12 identify the von Mises stresses at Z1 and Z2 for all elements in the web and in the plate. These responses are, in turn, constrained by the bounds set using a corresponding set of DCONSTR entries (numbers 3, 6, 9, and 12).

From the summary of design cycle history, note that the initial constraint violation is greater than 100%. Even though the initial design is infeasible, the optimizer is able to obtain a feasible design after the second iteration. However, this is at the expense of additional structural mass with a weight increase to 16.4 lbs. from 5.8 lbs. Once a feasible design is achieved, the optimizer can begin trying to reduce the weight, achieving a final weight of 7.9 lbs. Of course, this design is still heavier than the original structure (by almost 40%), yet that design did not satisfy the performance constraints whereas this one does.

```
***** NORMAL CONVERGENCE CRITERIA SATISFIED ***** (HARD CONVERGENCE DECISION LOGIC)
**************************************************************************************
                 CONVERGENCE ACHIEVED BASED ON THE FOLLOWING CRITERIA
                          (HARD CONVERGENCE DECISION LOGIC)
       RELATIVE CHANGE IN OBJECTIVE        0.0000E+00  MUST BE LESS THAN   1.0000E-03
 OR    ABSOLUTE CHANGE IN OBJECTIVE        0.0000E+00  MUST BE LESS THAN   1.0000E-02
                                   --- AND ---
       MAXIMUM CONSTRAINT VALUE            1.8636E-03  MUST BE LESS THAN   5.0000E-03
                         (CONVERGENCE TO A FEASIBLE DESIGN)
                                   --- OR ---
       MAXIMUM OF RELATIVE PROP. CHANGES   0.0000E+00  MUST BE LESS THAN   1.0000E-03
 AND   MAXIMUM OF RELATIVE D.V. CHANGES    0.0000E+00  MUST BE LESS THAN   1.0000E-03
                (CONVERGENCE TO A BEST COMPROMISE INFEASIBLE DESIGN)
**************************************************************************************
```

```
                 ***************************************************************
                 S U M M A R Y   O F   D E S I G N   C Y C L E   H I S T O R Y
                 ***************************************************************
                                   (HARD CONVERGENCE ACHIEVED)
                                   (SOFT CONVERGENCE ACHIEVED)
                     NUMBER OF FINITE ELEMENT ANALYSES COMPLETED           8
                     NUMBER OF OPTIMIZATIONS W.R.T. APPROXIMATE MODELS     7
                            OBJECTIVE AND MAXIMUM CONSTRAINT HISTORY
----------------------------------------------------------------------------------------------------------------
               OBJECTIVE FROM       OBJECTIVE FROM       FRACTIONAL ERROR      MAXIMUM VALUE         ROW OF
  CYCLE         APPROXIMATE            EXACT                   OF                   OF               MAXIMUM
  NUMBER       OPTIMIZATION           ANALYSIS           APPROXIMATION          CONSTRAINT          CONSTRAINT
----------------------------------------------------------------------------------------------------------------
  INITIAL                           5.784520E+00                               1.311878E+00            10
      1        1.122267E+01        1.122355E+01       -7.868297E-05         4.089278E-01             2
      2        1.636887E+01        1.636941E+01       -3.285838E-05        -1.610547E-03             2
      3        1.294620E+01        1.294636E+01       -1.259646E-05        -8.604820E-02             1
      4        8.748927E+00        8.751337E+00       -2.753791E-04         1.988992E-02            10
      5        7.379158E+00        7.379156E+00        3.230973E-07         9.885500E-02            21
      6        7.948771E+00        7.948517E+00        3.197505E-05         1.863594E-03            19
      7        7.948491E+00        7.948517E+00       -3.359479E-06         1.863594E-03            19
----------------------------------------------------------------------------------------------------------------
```

```
                                                  DESIGN VARIABLE HISTORY
------------------------------------------------------------------------------------------------------------------------------
  INTERNAL |   EXTERNAL    |             |
  DV. ID.  |    DV. ID.    |    LABEL    |   INITIAL    :      1       :      2       :     3       :      4       :     5      :
------------------------------------------------------------------------------------------------------------------------------
        1  |          1    |  T-PLATE    |   1.5000E-01 :   3.0000E-01 :   4.4321E-01 :  2.8106E-01 :   1.4038E-01 : 1.0879E-01 :
        2  |          2    |  T-WEB      |   2.0000E-01 :   4.0000E-01 :   6.1265E-01 :  8.4962E-01 :   8.2809E-01 : 7.5169E-01 :
        3  |          3    |  A-BAR      |   1.4400E-01 :   1.6592E-01 :   1.2682E-01 :  6.4814E-02 :   3.2407E-02 : 1.6165E-02 :
------------------------------------------------------------------------------------------------------------------------------
  INTERNAL |   EXTERNAL    |             |
  DV. ID.  |    DV. ID.    |    LABEL    |        6     :        7     :         8    :        9    :       10     :        11 :
------------------------------------------------------------------------------------------------------------------------------
        1  |          1    |  T-PLATE    |   1.1180E-01 :   1.1180E-01 :
        2  |          2    |  T-WEB      |   8.3723E-01 :   8.3723E-01 :
        3  |          3    |  A-BAR      |   1.6165E-02 :   1.6165E-02 :
------------------------------------------------------------------------------------------------------------------------------
 *** USER INFORMATION MESSAGE 6464 (DOM12E)
     RUN TERMINATED DUE TO HARD CONVERGENCE TO A UNIQUE OPTIMUM AT CYCLE NUMBER =        7.
```

```
ID UGS, D200X7
TIME  10
SOL 200       $  OPTIMIZATION
CEND
$
TITLE  = STATIC ANALYSIS OF A STIFFENED PLATE
ECHO   = UNSORT
DISP   = ALL
STRESS = ALL
SPC    = 1
ANALYSIS = STATICS
DESOBJ(MIN) = 15     $ OBJECTIVE FUNCTION DEFINITION
$                      (MIN IS THE DEFAULT)
SUBCASE 1
   LABEL  = LOAD CONDITION 1
   LOAD   = 1
   DESSUB = 100       $ CONSTRAINT DEFININITION
SUBCASE 2
   LABEL  = LOAD CONDITION 2
   LOAD   = 2
   DESSUB = 200       $ CONSTRAINT DEFININITION
BEGIN BULK
$
$-----------------------------------------------------------------------
$ ANALYSIS MODEL:
$-----------------------------------------------------------------------
$
GRID,   10000,  ,       0.0,    0.0,    0.0
GRID,   10001,  ,       2.5,    0.0,    0.0
GRID,   10002,  ,       5.0,    0.0,    0.0
GRID,   10003,  ,       7.5,    0.0,    0.0
GRID,   10004,  ,       10.0,   0.0,    0.0
GRID,   10100,  ,       0.0,    2.5,    0.0
GRID,   10101,  ,       2.5,    2.5,    0.0
GRID,   10102,  ,       5.0,    2.5,    0.0
GRID,   10103,  ,       7.5,    2.5,    0.0
GRID,   10104,  ,       10.0,   2.5,    0.0
GRID,   10200,  ,       0.0,    5.0,    0.0
GRID,   10201,  ,       2.5,    5.0,    0.0
GRID,   10202,  ,       5.0,    5.0,    0.0
GRID,   10203,  ,       7.5,    5.0,    0.0
GRID,   10204,  ,       10.0,   5.0,    0.0
GRID,   10300,  ,       0.0,    7.5,    0.0
GRID,   10301,  ,       2.5,    7.5,    0.0
GRID,   10302,  ,       5.0,    7.5,    0.0
GRID,   10303,  ,       7.5,    7.5,    0.0
GRID,   10304,  ,       10.0,   7.5,    0.0
GRID,   10400,  ,       0.0,    10.0,   0.0
GRID,   10401,  ,       2.5,    10.0,   0.0
GRID,   10402,  ,       5.0,    10.0,   0.0
GRID,   10403,  ,       7.5,    10.0,   0.0
GRID,   10404,  ,       10.0,   10.0,   0.0
GRID,   20100,  ,       0.0,    5.0,    1.0
GRID,   20101,  ,       2.5,    5.0,    1.0
GRID,   20102,  ,       5.0,    5.0,    1.0
GRID,   20103,  ,       7.5,    5.0,    1.0
GRID,   20104,  ,       10.0,   5.0,    1.0
GRID,   20200,  ,       0.0,    5.0,    2.0
GRID,   20201,  ,       2.5,    5.0,    2.0
GRID,   20202,  ,       5.0,    5.0,    2.0
GRID,   20203,  ,       7.5,    5.0,    2.0
GRID,   20204,  ,       10.0,   5.0,    2.0
$
CQUAD4,   1,    1,      10000,  10001,  10101,  10100
CQUAD4,   2,    1,      10001,  10002,  10102,  10101
CQUAD4,   3,    1,      10002,  10003,  10103,  10102
CQUAD4,   4,    1,      10003,  10004,  10104,  10103
CQUAD4,   5,    1,      10100,  10101,  10201,  10200
CQUAD4,   6,    1,      10101,  10102,  10202,  10201
CQUAD4,   7,    1,      10102,  10103,  10203,  10202
CQUAD4,   8,    1,      10103,  10104,  10204,  10203
CQUAD4,   9,    1,      10200,  10201,  10301,  10300
CQUAD4,  10,    1,      10201,  10202,  10302,  10301
CQUAD4,  11,    1,      10202,  10203,  10303,  10302
CQUAD4,  12,    1,      10203,  10204,  10304,  10303
CQUAD4,  13,    1,      10300,  10301,  10401,  10400
CQUAD4,  14,    1,      10301,  10302,  10402,  10401
CQUAD4,  15,    1,      10302,  10303,  10403,  10402
CQUAD4,  16,    1,      10303,  10304,  10404,  10403
$
CQUAD4, 21,     2,      10200,  10201,  20101,  20100
CQUAD4, 22,     2,      10201,  10202,  20102,  20101
CQUAD4, 23,     2,      10202,  10203,  20103,  20102
CQUAD4, 24,     2,      10203,  10204,  20104,  20103
CQUAD4, 25,     2,      20100,  20101,  20201,  20200
CQUAD4, 26,     2,      20101,  20102,  20202,  20201
CQUAD4, 27,     2,      20102,  20103,  20203,  20202
CQUAD4, 28,     2,      20103,  20104,  20204,  20203
$
CBAR,   31,     3,      20200,  20201,  0.0,    1.0,    0.0
CBAR,   32,     3,      20201,  20202,  0.0,    1.0,    0.0
CBAR,   33,     3,      20202,  20203,  0.0,    1.0,    0.0
```

```
CBAR,   34,     3,      20203,  20204,  0.0,    1.0,    0.0
$
PSHELL, 1,      1,      0.15,   1
PSHELL, 2,      1,      0.2,    1
PBAR,   3,      1,      0.144,  1.728-4,1.728-2,1.745-2,,       ,       +PB3
+PB3,   0.06,   0.6,    0.06,   -0.6,   -0.06,  -0.6,   -0.06,  0.6
$
MAT1,   1,      1.0E+7, ,       0.33,   0.283
$
FORCE,  1,      10004,  ,       2000.0, 1.0,    0.0,    0.0
FORCE,  1,      10104,  ,       2000.0, 1.0,    0.0,    0.0
FORCE,  1,      10204,  ,       2000.0, 1.0,    0.0,    0.0
FORCE,  1,      10304,  ,       2000.0, 1.0,    0.0,    0.0
FORCE,  1,      10404,  ,       2000.0, 1.0,    0.0,    0.0
FORCE,  2,      10203,  ,       10000.0,0.0,    0.0,    1.0
PLOAD2, 1,      50.,    1,      THRU,   16
$
SPC1,   1,      1236,   10000
SPC1,   1,      136,    10100,  10300,  10400
SPC1,   1,      36,     10001,  10002,  10003,  10004,  10104
SPC1,   1,      36,     10401,  10402,  10403,  10404,  10304
SPC1,   1,      3,      10204
SPC1,   1,      13,     10200
SPC1,   1,      5,      20100,  20101,  20102,  20103,  20104
SPC1,   1,      6,      10101,  10102,  10103,  10104
SPC1,   1,      6,      10301,  10302,  10303,  10304
$
PARAM,  GRDPNT,  1
PARAM,  WTMASS,  0.00259
PARAM,  AUTOSPC, YES
$
$-------------------------------------------------------------------------
$ DESIGN MODEL:
$-------------------------------------------------------------------------
$
$...Define the design variables:
$
$DESVAR,ID,     LABEL,  XINIT,  XLB,    XUB,    DELXV
DESVAR, 1,      T-PLATE,0.15,   0.001,  10.0
DESVAR, 2,      T-WEB,  0.20,   0.001,  10.0
DESVAR, 3,      A-BAR,  0.144,  0.001,  10.0
$
$...Relate the design variables to analysis model properties
$   (linear relations, so use DVPREL1)
$
$...Express shell thicknesses as functions of x1, x2:
$DVPREL1,ID,    TYPE,   PID,    FID,    PMIN,   PMAX,   C0,     ,       +
$+,     DVIDD1, COEF1,  DVID2,  COEF2,  ...
DVPREL1,1,      PSHELL, 1,      4,      0.01,   ,       ,       ,       +DP1
+DP1,   1,      1.0
DVPREL1,2,      PSHELL, 2,      4,      0.01,   ,       ,       ,       +DP2
+DP2,   2,      1.0
$
$...Express bar cross sectional area as a function of x3:
DVPREL1,3,      PBAR,   3,      4,      0.01,   ,       ,       ,       +DP3
+DP3,   3,      1.0
$
$...Proportionally relate bar's I1, I2, I12 to changes in area (x3):
$   (nonlinear relations require use of DVPREL2+DEQATN)
$DVPREL2,ID,    TYPE,   PID,    FID,    PMIN,   PMAX,   EQID,   ,       +
$+,      DESVAR,DVID1,  DVID2,  ...,    ,       ,       ,       ,       +
$+,      DTABLE,CID1,   CID2,   ...
DVPREL2,11,     PBAR,   3,      5,      1.0-6,  ,       101,    ,       +
+,      DESVAR, 3,      ,       ,       ,       ,       ,       ,       +
+,      DTABLE, X3INIT, I1INIT
DEQATN  101     I1(X3,X3INIT,I1INIT) = I1INIT          ;
                               DELTA = SQRT(X3/X3INIT) ;
                               I1NEW = I1*DELTA**4
DVPREL2,12,     PBAR,   3,      6,      1.0-6,  ,       102,    ,       +
+,      DESVAR, 3,      ,       ,       ,       ,       ,       ,       +
+,      DTABLE, X3INIT, I2INIT
DEQATN  102     I2(X3,X3INIT,I2INIT) = I2INIT          ;
                               DELTA = SQRT(X3/X3INIT) ;
                               I2NEW = I2*DELTA**4
DVPREL2,13,     PBAR,   3,      7,      1.0-6,  ,       103,    ,       +
+,      DESVAR, 3,      ,       ,       ,       ,       ,       ,       +
+,      DTABLE, X3INIT, I12INIT
DEQATN  103     I12(X3,X3INIT,I12INIT) = I12INIT          ;
                                DELTA  = SQRT(X3/X3INIT)  ;
                                I12NEW = I12*DELTA**4
$
$...Modify stress recovery points accordingly:
DVPREL2,14,     PBAR,   3,      12,     ,       ,       104,    ,       +
+,      DESVAR, 3,      ,       ,       ,       ,       ,       ,       +
+,      DTABLE, X3INIT, CYINIT
DVPREL2,15,     PBAR,   3,      13,     ,       ,       104,    ,       +
+,      DESVAR, 3,      ,       ,       ,       ,       ,       ,       +
+,      DTABLE, X3INIT, CZINIT
DVPREL2,16,     PBAR,   3,      14,     ,       ,       104,    ,       +
+,      DESVAR, 3,      ,       ,       ,       ,       ,       ,       +
+,      DTABLE, X3INIT, DYINIT
DVPREL2,17,     PBAR,   3,      15,     -1.0,   ,       104,    ,       +
+,      DESVAR, 3,      ,       ,       ,       ,       ,       ,       +
+,      DTABLE, X3INIT, DZINIT
DVPREL2,18,     PBAR,   3,      16,     -1.0,   ,       104,    ,       +
+,      DESVAR, 3,      ,       ,       ,       ,       ,       ,       +
+,      DTABLE, X3INIT, EYINIT
DVPREL2,19,     PBAR,   3,      17,     -1.0,   ,       104,    ,       +
+,      DESVAR, 3,      ,       ,       ,       ,       ,       ,       +
+,      DTABLE, X3INIT, EZINIT
DVPREL2,20,     PBAR,   3,      18,     -1.0,   ,       104,    ,       +
+,      DESVAR, 3,      ,       ,       ,       ,       ,       ,       +
+,      DTABLE, X3INIT, FYINIT
DVPREL2,21,     PBAR,   3,      19,     ,       ,       104,    ,       +
+,      DESVAR, 3,      ,       ,       ,       ,       ,       ,       +
+,      DTABLE, X3INIT, FZINIT
$
$...Equation for stress recovery points:
DEQATN  104     NEWPOINT(X3,X3INIT,POINT) = POINT*SQRT(X3/X3INIT)
$
$...Table constants for all equations:
DTABLE, X3INIT, 0.144,  I1INIT, 1.728-4,I2INIT, 1.728-2,I12INIT,1.745-2,+
+,      CYINIT, 0.06,   CZINIT, 0.6,    DYINIT, 0.06,   DZINIT, -0.6,   +
+,      EYINIT, -0.06,  EZINIT, -0.6,   FYINIT, -0.06,  FZINIT, 0.6
$
$...Identify the design responses:
$
$DRESP1,ID,     LABEL,  RTYPE,  PTYPE,  REGION, ATTA,   ATTB,   ATT1,   +
$+,     ATT2,   ...
DRESP1,  1,     SBARA,  STRESS, PBAR,   ,       7,      ,       3
DRESP1,  2,     SBARB,  STRESS, PBAR,   ,       14,     ,       3
DRESP1,  3,     S13,    STRESS, PSHELL, ,       9,      ,       1
DRESP1,  6,     S16,    STRESS, PSHELL, ,       17,     ,       1
DRESP1,  9,     S23,    STRESS, PSHELL, ,       9,      ,       2
DRESP1, 12,     S26,    STRESS, PSHELL, ,       17,     ,       2
DRESP1, 13,     D1,     DISP,   ,       ,       3,      ,       10302
DRESP1, 14,     D2,     DISP,   ,       ,       3,      ,       10203
DRESP1, 15,     W,      WEIGHT
$
$...Place bounds on the responses:
$
$DCONSTR,DCID,  RID,    LALLOW, UALLOW
DCONSTR,  10,   1,      -25000.,25000.
DCONSTR,  10,   2,      -25000.,25000.
DCONSTR,  10,   3,      -25000.,25000.
DCONSTR,  10,   6,      -25000.,25000.
DCONSTR,  10,   9,      -25000.,25000.
DCONSTR,  10,   12,     -25000.,25000.
DCONSTR,  20,   13,     -0.1,   0.1
DCONSTR,  30,   14,     -0.03,  0.03
$
$DCONADD,DCID,  DC1,    DC2,    ...
DCONADD,  100,  10,     20      $ summed constraint set for subcase 1
DCONADD,  200,  10,     30      $ summed constraint set for subcase 2
$
$...Optional override of optimization control parameters:
$
DOPTPRM,IPRINT, 1,      DESMAX, 20,     DELP,   0.5,    P1,     1,      +
+,      P2,     15
$ (DELP=0.5 allows larger moves, thus overcoming constraint
$  violations quicker)
$
ENDDATA
```

Listing 7-4.

7.5 Shape Optimization of a Culvert

This example considers shape optimization using the direct input of shapes method.

In order to use this method, sets of displacement vectors are first generated using a separate auxiliary model analysis. These displacement vectors are then DBLOCATEd in a subsequent design optimization run and used to generate a set of shape basis vectors. The optimizer then seeks to find the best combination of these shape basis vectors.

Problem Description

Figure 7-6 shows the finite element model of one-half of a symmetric culvert structure. (Symmetry exists with respect to the y-axis.) The structure is made of steel, and has been modeled using

CQUAD4 elements in plane strain. This example has been taken from A. D. Belegundu, and S. D. Rajan, “Shape Optimal Design Using Isoparametric Elements”, Proceedings of 29th AIAA/ASME/ASCE/AHS/ASC Structures, Dynamics, and Materials Conference, pp. 696–701, Williamsburg, VA, April 18–20, 1988.

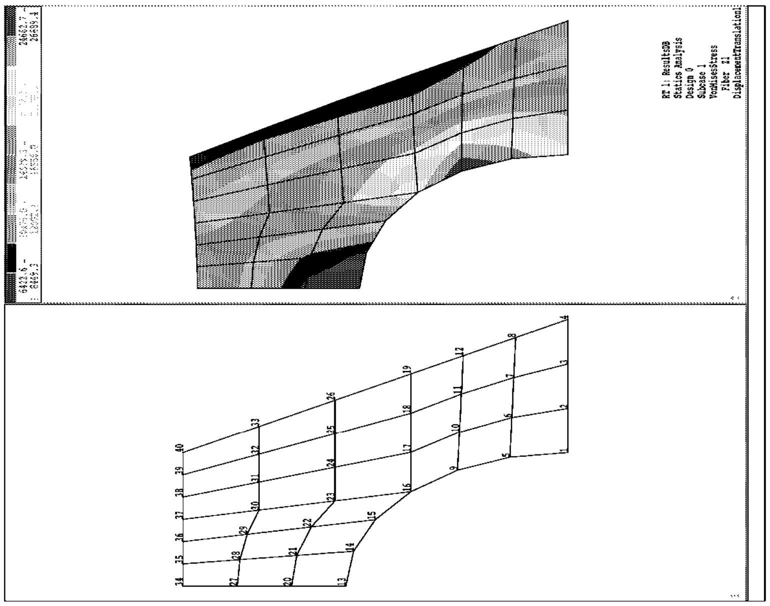


Figure 7-6. Initial Culvert Design

With the bottom surface fixed, pressure loads are applied on the top surface. The design task is to minimize the volume of the structure by changing the shape of the initially circular hole, subject to von Mises stress constraints over the interior.

Modeling Considerations

The process of shape basis vector generation is closely associated with the nature of the design problem itself. For example, since the goal here is weight minimization, the shape basis vectors must clearly be able to reduce the volume of the structure. Were they to simply redistribute material, the weight would not change and the optimizer would be able to make little progress.

Furthermore, since the redesign is subject to von Mises stresses, we can use the initial stress distribution to help us in the selection of appropriate basis vectors. This is also shown in Figure 7-6. Here we note that the maximum von Mises stresses occur around the lower portion of the circular hole. The stresses are not equally distributed, either. This tells us that a circular boundary does not provide the optimal design. Thus, our shape basis vectors must contain other than just radial components.

We also need to decide how many design variables (or shape basis vectors) are needed. The decision is not unique because the selection can be affected by experience, various functional and manufacturing requirements, or aesthetic requirements. However, the shape basis vectors should yield as much generality as possible, consistent with our design goals.

Creation of Auxiliary Model, and Generation of Basis Vectors

The top left frame in Figure 7-7 shows the auxiliary model geometry and boundary conditions. Note that it has the same geometry as the primary structure (Figure 7-6), but with different boundary and loading conditions. The input data file for this model is given in Listing 7-5.

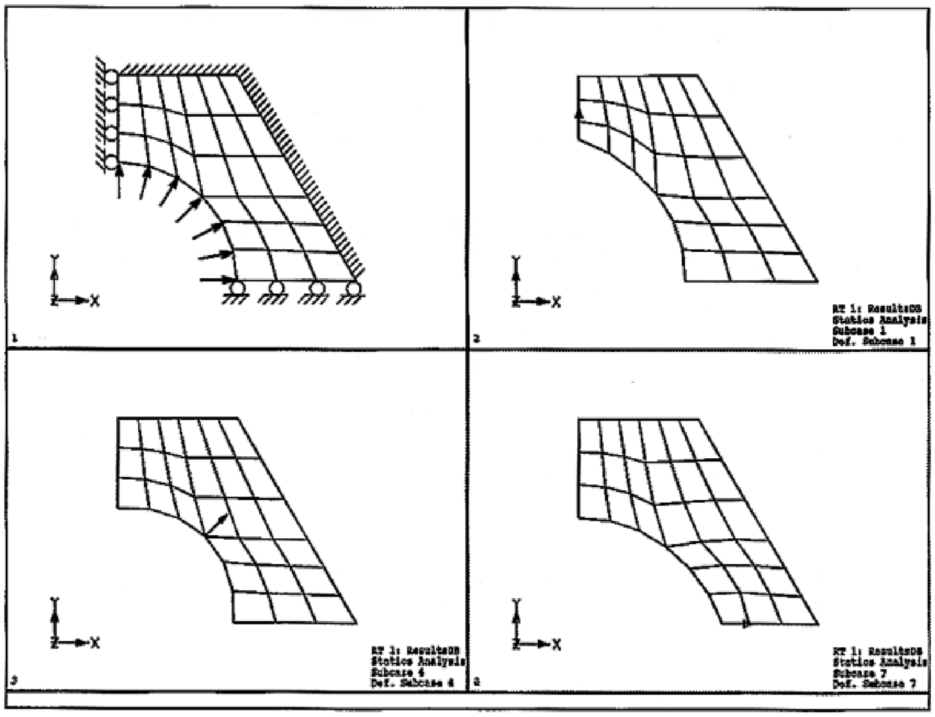


Figure 7-7. Auxiliary Model

Outside edges of the culvert are fixed in the auxiliary model to satisfy straight edge requirements. Grid points 1, 2, 3 on the bottom are allowed to move in the x-direction and grid points 13, 20, 27 on the symmetry line can move along the y-direction. Grids 5, 9, 14, 15, and 16 on the hole boundary are allowed to move as well. Along this hole boundary, six CBAR elements have been added to help smooth the applied loading effects. (It is important to allow z-rotations along this boundary.)

To generate a set of displacements to be used as basis vectors, we can statically load each grid in a direction normal to the boundary as shown in Figure 7-7. Note from Listing 7-5 that this can be performed in Solution 101 (or in Solution 200, with OPTEXIT = 2). This will result in seven displacement vectors, one corresponding to each load case. Three of these vectors are also shown in Figure 7-7. Each arrow on a plot indicates that the deformation is due to a concentrated force.

Design Optimization Input

In the optimization run, the displacement matrix containing the seven displacement vectors from the Auxiliary Model analysis is retrieved using the DBLOCATE statement. The optimization input file is shown in Listing 7-7.

Since each shape basis vector is defined in terms of a single displacement vector, seven DESVAR and DVSHAP entries are used to define seven shape basis vectors. The seven DVSHAP entry scaling factors have been selected such that the maximum component of each shape basis vector is unity. For example, DVSHAP entry number 1 identifies the first shape basis vector in terms of the first DBLOCATE'd displacement vector (1 in field 3). The 1 in field 2 indicates that design variable number 1 is to be the multiplier of this vector. Furthermore, since the maximum component of this vector is 1.0/66.773 (as determined from a manual inspection of the data), the scaling factor has been given as 66.773, effectively unit normalizing the vector.

Each design variable acts as a multiplying coefficient of a shape basis vector. Initial values are somewhat arbitrary and have been selected as 3.0 here. However, when combined with 20% allowable move limits (0.2 in field 7 of the DESVAR entries), it can be seen that part of the reason for the choice is that these initial values provide for reasonable move limits on the initial design. Lower and upper limits of -1.0E6 and +1.0E6 indicate that the design variables are to be considered effectively unbounded during optimization.

The final shape after six design cycles is plotted in Figure 7-8 (bottom half) with the corresponding stress contour plot of the deformed structure (top half). Note the favorable distribution of von Mises stresses for the final shape. The volume of the culvert has been reduced by about 20% while the strength constraint is satisfied.

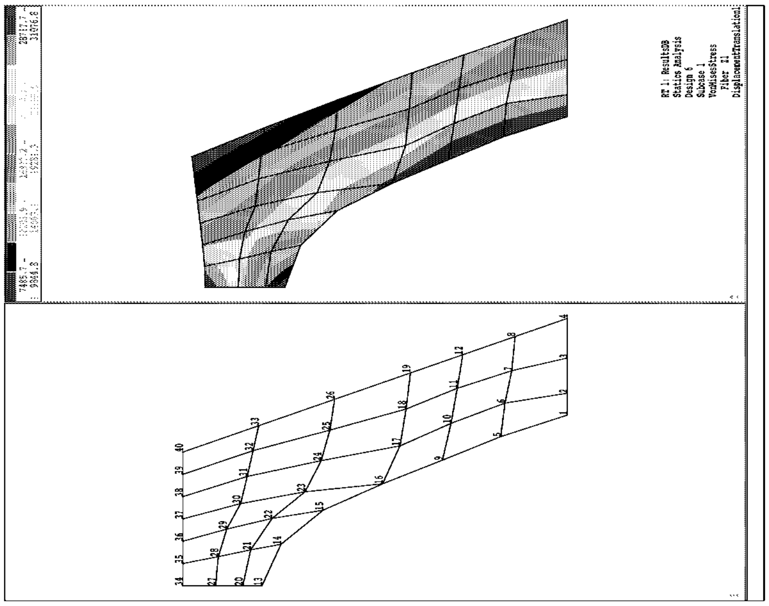


Figure 7-8. Final Culvert Design

```
ID,AUX1,VT100 $FEB 10,1990
TIME 10
SOL 101
CEND
TITLE=Culvert Example Using External Auxiliary Model
SUBTITLE=The External Auxiliary Model
SPC=25
$
$ seven load cases
$
SUBCASE 1
 LOAD=100
 DISP=ALL
SUBCASE 2
 LOAD=101
 DISP=ALL
SUBCASE 3
 LOAD=102
 DISP=ALL
SUBCASE 4
 LOAD=103
 DISP=ALL
SUBCASE 5
 LOAD=104
 DISP=ALL
SUBCASE 6
 LOAD=105
 DISP=ALL
SUBCASE 7
 LOAD=106
 DISP=ALL
BEGIN BULK
PARAM,POST,0
param,newseq,-1
$
$ The same GRID and CQUAD4 entries as the primary structure
$
GRID,  1,, 3.00000, 0.00000,.00
GRID,  2,, 4.00000, 0.00000,.00
GRID,  3,, 5.00000, 0.00000,.00
  .    .      .        .      .
  .    .      .        .      .
 (see optimization input file)
  .    .      .        .      .
  .    .      .        .      .
GRID, 38,, 2.00000, 5.19600,.00
GRID, 39,, 2.50000, 5.19600,.00
GRID, 40,, 3.00000, 5.19600,.00
CQUAD4,    1,101,    1,    2,    6,    5
CQUAD4,    2,101,    2,    3,    7,    6
CQUAD4,    3,101,    3,    4,    8,    7
  .        .  .      .     .     .     .
  .        .  .      .     .     .     .
 (see optimization input file)
  .        .  .      .     .     .     .
  .        .  .      .     .     .     .
CQUAD4,   25,101,   30,   31,   38,   37
CQUAD4,   26,101,   31,   32,   39,   38
CQUAD4,   27,101,   32,   33,   40,   39
PSHELL,101,102,.44
MAT1,102,2.+7,,.3
$
$ Additional CBAR elements maintain smoothness of the circular boundary
$
CBAR,31,1,13,14,,1.0
CBAR,32,1,14,15,,1.0
CBAR,33,1,15,16,,1.0
CBAR,34,1,16, 9,,1.0
CBAR,35,1, 9, 5,,1.0
CBAR,36,1,5 , 1,,1.0
PBAR    1       102     20.0    1.0     1.0
$
$ Seven load cases
$
FORCE,100,13,0,1.e5,0.,1.,0.
FORCE,101,14,0,1.e5,0.259,.9659
FORCE,102,15,0,1.e5,0.5,0.866,0.0
FORCE,103,16,0,1.e5,1.,1.,0.
FORCE,104,9,0,1.e5,0.866,0.5,0.0
FORCE,105,5,0,1.e5,0.9659,0.259
FORCE,106,1,0,1.e5,1.,0.,0.
$
$ Boundary conditions satisfy functional and manufacturing requirements
$
SPC1,25,345,1,THRU,40
SPC1,25,6,2,THRU,4
SPC1,25,6,6,THRU,8
SPC1,25,6,10,THRU,12
SPC1,25,6,17,THRU,19
SPC1,25,6,20,THRU,26
SPC1,25,6,27,THRU,33
SPC1,25,6,34,THRU,40
SPC1,25,12,33,THRU,40
SPC1,25,12,4,8,12,19,26
SPC1,25,1,13,20,27
SPC1,25,2,1,2,3
ENDDATA
```

Listing 7-5.

```
$
$ FMS section for retrieving the auxiliary displacement matrix
$
assign f1_aux='culvert1.MASTER'
dblocate datablk=(ug/ugd,geom1/geom1d,geom2/geom2d) ,
 logical=f1_aux
SOL    200  $
TIME   100
CEND
TITLE=CULVERT EXAMPLE USING EXTERNAL AUXILIARY STRUCTURE
SUBTITLE=THE PRIMARY STRUCTURE
ANALYSIS = STATICS
SPC=25
 LOAD=1
 DISP=ALL
 STRESS=all
 DESSUB = 10
 desobj = 5
BEGIN BULK
PARAM,POST,0
PARAM,CDIF,NO
$ PARAM,optexit,4
PARAM,NEWSEQ,-1
GRID,  1,, 3.00000, 0.00000,.00
GRID,  2,, 4.00000, 0.00000,.00
GRID,  3,, 5.00000, 0.00000,.00
GRID,  4,, 6.00000, 0.00000,.00
GRID,  5,, 2.89464, 0.78478,.00
GRID,  6,, 3.79369, 0.75885,.00
GRID,  7,, 4.69274, 0.73293,.00
GRID,  8,, 5.59178, 0.70700,.00
GRID,  9,, 2.60164, 1.49178,.00
GRID, 10,, 3.46229, 1.46585,.00
GRID, 11,, 4.32293, 1.43993,.00
GRID, 12,, 5.18357, 1.41400,.00
GRID, 13,, 0.00000, 3.00000,.00
GRID, 14,, 0.78478, 2.89464,.00
GRID, 15,, 1.49178, 2.60164,.00
GRID, 16,, 2.12100, 2.12100,.00
GRID, 17,, 3.00578, 2.12100,.00
GRID, 18,, 3.89057, 2.12100,.00
GRID, 19,, 4.77535, 2.12100,.00
GRID, 20,, 0.00000, 3.73200,.00
GRID, 21,, 0.68985, 3.66176,.00
GRID, 22,, 1.32785, 3.46643,.00
GRID, 23,, 1.91400, 3.14600,.00
GRID, 24,, 2.67052, 3.14600,.00
GRID, 25,, 3.42704, 3.14600,.00
GRID, 26,, 4.18357, 3.14600,.00
GRID, 27,, 0.00000, 4.46400,.00
GRID, 28,, 0.59493, 4.42888,.00
GRID, 29,, 1.16393, 4.33122,.00
GRID, 30,, 1.70700, 4.17100,.00
GRID, 31,, 2.33526, 4.17100,.00
GRID, 32,, 2.96352, 4.17100,.00
GRID, 33,, 3.59178, 4.17100,.00
GRID, 34,, 0.00000, 5.19600,.00
GRID, 35,, 0.50000, 5.19600,.00
GRID, 36,, 1.00000, 5.19600,.00
GRID, 37,, 1.50000, 5.19600,.00
GRID, 38,, 2.00000, 5.19600,.00
GRID, 39,, 2.50000, 5.19600,.00
GRID, 40,, 3.00000, 5.19600,.00
CQUAD4,    1,101,    1,    2,    6,    5
CQUAD4,    2,101,    2,    3,    7,    6
CQUAD4,    3,101,    3,    4,    8,    7
CQUAD4,    4,101,    5,    6,   10,    9
CQUAD4,    5,101,    6,    7,   11,   10
CQUAD4,    6,101,    7,    8,   12,   11
CQUAD4,    7,101,    9,   10,   17,   16
CQUAD4,    8,101,   10,   11,   18,   17
CQUAD4,    9,101,   11,   12,   19,   18
CQUAD4,   10,101,   13,   14,   21,   20
CQUAD4,   11,101,   14,   15,   22,   21
CQUAD4,   12,101,   15,   16,   23,   22
CQUAD4,   13,101,   20,   21,   28,   27
CQUAD4,   14,101,   21,   22,   29,   28
CQUAD4,   15,101,   22,   23,   30,   29
CQUAD4,   16,101,   27,   28,   35,   34
CQUAD4,   17,101,   28,   29,   36,   35
CQUAD4,   18,101,   29,   30,   37,   36
CQUAD4,   19,101,   16,   17,   24,   23
CQUAD4,   20,101,   17,   18,   25,   24
CQUAD4,   21,101,   18,   19,   26,   25
CQUAD4,   22,101,   23,   24,   31,   30
CQUAD4,   23,101,   24,   25,   32,   31
CQUAD4,   24,101,   25,   26,   33,   32
CQUAD4,   25,101,   30,   31,   38,   37
CQUAD4,   26,101,   31,   32,   39,   38
CQUAD4,   27,101,   32,   33,   40,   39
FORCE   1       34      0       1250.           -1.
FORCE   1       35      0       2500.           -1.
FORCE   1       36      0       2500.           -1.
FORCE   1       37      0       2500.           -1.
FORCE   1       38      0       2500.           -1.
FORCE   1       39      0       2500.00         -1.
FORCE   1       40      0       1250.           -1.
PSHELL,101,102,.44
MAT1,102,2.+7,,.3,0.731-3
SPC1,25,3456,1,THRU,40
SPC1,25,12,1,THRU,4
SPC1,25,1,13,20,27,34
$
$ design model
$
desvar  1       b1      3.      -1.e6      1.e6 .2
desvar  2       b1      3.      -1.e6      1.e6 .2
desvar  3       b3      3.      -1.e6      1.e6 .2
desvar  4       b4      3.      -1.e6      1.e6 .2
desvar  5       b5      3.      -1.e6      1.e6 .2
desvar  6       b6      3.      -1.e6      1.e6 .2
desvar  7       b7      3.      -1.e6      1.e6 .2
$
$ A DVSHAP entry defines a shape basis vector by associating one design
$ variable to a dblocated displacement vector.
$
dvshap  1       1       66.773
dvshap  2       2       117.35
dvshap  3       3       216.33
dvshap  4       4       443.55
dvshap  5       5       220.89
dvshap  6       6       115.69
dvshap  7       7       65.669
dresp1  5       volume  volume
dresp1  2       von-mis stress  pshell          9               101
DCONSTR 10      2       -3.100e43.100e4
doptprm DESMAX  25      APRCOD  1
param,nasprt,1
ENDDATA
```

Listing 7-6.

7.6 Analytic Boundary Shapes

This example illustrates the use of the analytic boundary shapes method in shape optimal design. Relating Design Variables to Shape Changes includes a checklist for setting up the design model using this method. You may want to refer to that section in connection with this example.

To use this method, you need to define auxiliary models over the boundaries of the structure. When constrained and loaded, these boundary models produce static deformations that can be used to describe shape variations over the boundaries. The code then interpolates this information to the interior grids, resulting in basis vectors for shape optimization. A static analysis is used for this interpolation.

Problem Description

Figure 7-9 shows the initial structure. It is a simple cantilever, modeled with eighty solid elements, fixed at the support and tip-loaded at the free end. The design goal is to minimize the structure's weight subject to constraints on von Mises stresses. We'll investigate minimizing the weight by tapering the cantilever's shape. The initial stress distribution is shown in Figure 7-10.

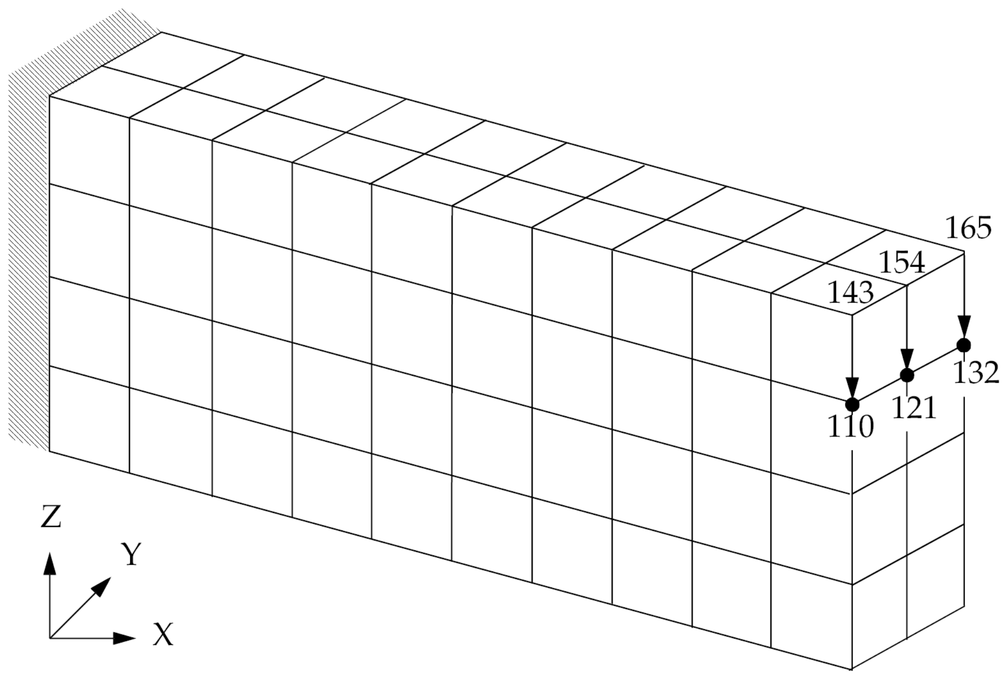


Figure 7-9. Solid Cantilever

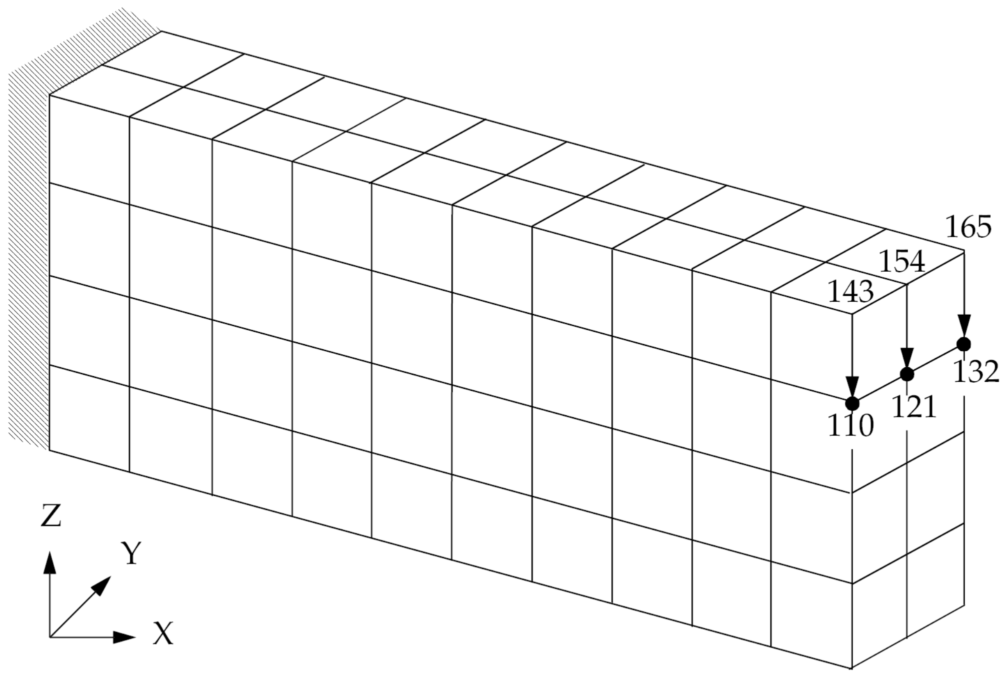


Figure 7-10. Solid Cantilever - Initial Stress Distribution

Since the cantilever is oriented and loaded in an x-z plane, removing material from the upper and lower surfaces is an effective way to reduce the weight, tapering the cantilever from its root to its tip. Basis vectors describing this characteristic shape can be easily generated using the analytic boundary shapes method.

Boundary Shape Changes Using Auxiliary Boundary Models

Figure 7-11 shows the auxiliary boundary models that can be used to generate these shapes. These disjoint models, one for the upper surface and one for the lower, are built using QUAD4 elements. When fixed at the root and loaded at the tip, each generates a cubic displacement variation over its length.

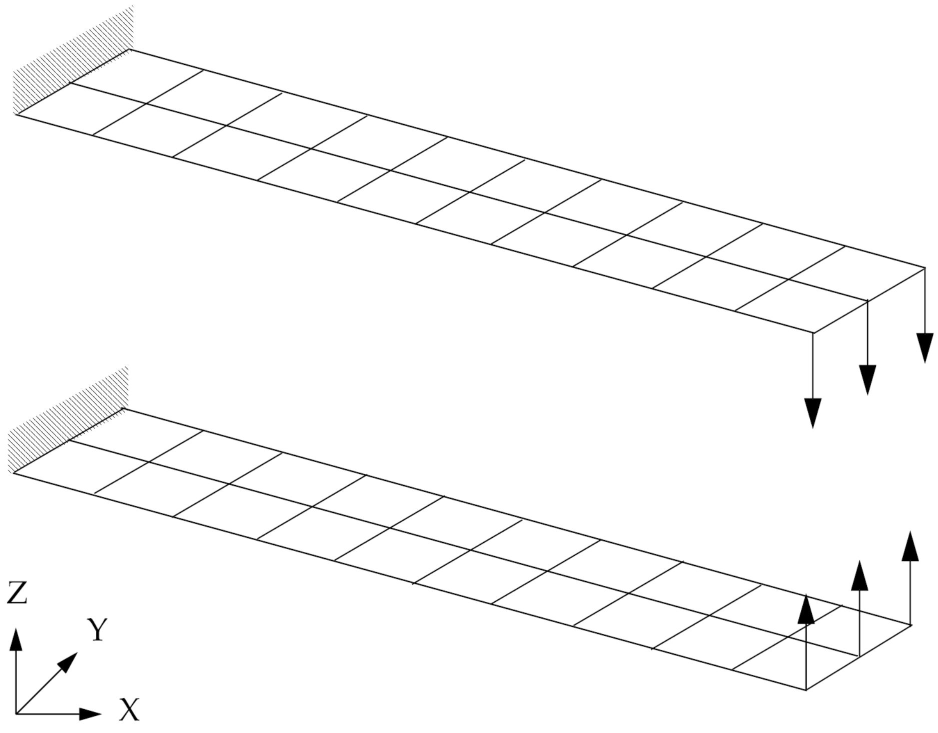


Figure 7-11. Auxiliary Boundary Models

Listing 7-8 is an abbreviated input file for the combined analysis and optimization of the primary structure, and the auxiliary boundary model specification. The complete file can be found in the Test Problem Library as D200AM3.DAT. The primary model definition is similar to that of other NX Nastran input files for static analysis shown in this guide, so its description will be deferred here. The optimization-related input for the primary structure will be described shortly.

Turning first to the auxiliary boundary model specification, we see that these boundary models are defined in a special Bulk Data Section, appearing after the Bulk Data for the primary model. The statement, BEGIN BULK AUXMODEL = 1, is used to indicate the beginning of this section. This section essentially defines three components of the model: its connectivity, its loads, and its boundary conditions. This is just like any other NX Nastran model for static analysis.

The auxiliary boundary model connectivity is defined using CQUAD4 elements. The geometry should not be redefined, since the primary model geometry is used. (If necessary, additional grids can be included. For example, an additional grid may be necessary for use as a rigid element connection point. This grid should then appear in this Bulk Data Section.)

The structure is constrained using two SPC sets, 200 and 300. Each set, one for each of the two auxiliary model subcases, is coupled with one of two separate enforced displacement type loading conditions: an enforced tip displacement in the +z direction for the lower surface and an enforced tip displacement in the -z direction for the upper surface. These are defined using SPCD sets 220 and 330, respectively.

The Case Control for these models appears after the primary model Case Control, in a section beginning with the label, AUXCASE. Each subcase selects one of the two SPCD-defined loads, using a different boundary condition for each. Static loading is always assumed for these auxiliary model analyses.

The results of these two analyses are sets of displacement vectors, which essentially interpolate the tip displacement along the length of each "plate." This smooth shape interpolation is now quite useful for describing shape changes over the Primary Structure's boundary. However, these boundary displacements must still be interpolated over the structure's interior in order to form shape basis vectors for the entire structure. This is achieved in the primary model section of the Bulk Data.

Shape Changes over the Interior

BNDGRID entries provide the boundary conditions for the shape interpolation steps. The relation between BNDGRID entries and the auxiliary boundary model solutions is similar to that of SPC's and SPCD's. The auxiliary boundary model solutions are imposed as enforced displacements on the primary structure. This is like an SPCD. These displacement degrees of freedom must be present on BNDGRID entries, as with SPC entries. Degrees of freedom listed on the BNDGRID entries are either enforced or fixed, depending on whether or not an auxiliary boundary model solution exists for them. All other degrees of freedom are considered "free," and are solved for in the interpolation.

For example, grids 143, 154, and 165 are located at the tip of the upper surface (see Figure 7-9). Their displacement components are determined from the auxiliary model analysis. This is indicated by using a BNDGRID entry with a translational component of 123 given for each of these grids (see listing 7–7).

In contrast, the displacements at grids 110, 121, and 132, located directly below the previous set of grids, must be solved for during the interpolation phase. Since we want to allow the tip of the beam to narrow, the z displacement component must not be fixed. In contrast, all must be fixed in the x-direction (no axial shortening), and the corner grids (110 and 132) must be restrained in the y-direction (no shortening of the beam width). This leads to BNDGRID-defined components of 12 for grids 110 and 132, and component 1 only for grid 121.

Modeling Summary

To summarize, auxiliary boundary models are created using additional Bulk Data Sections, labeled using BEGIN BULK AUXMODEL = n. We can have as many of these sections as necessary to suit our auxiliary boundary model needs. Boundary conditions and loads are applied with AUXCASE-labelled Case Control Sections. The resultant boundary deformations are then interpolated to the interior of the primary structure using BNDGRID entries to define the boundaries. The resulting total displacement vectors can now be combined in any way to yield basis vectors for shape optimization.

Shape Basis Vectors

The shape basis vectors are defined using DVBSHAP Bulk Data entries. DVBSHAP entries 1 and 2 relate design variables 1 and 2, respectively, to the two displacement solutions by the relation:

$$[T] = \left[1.0U_{SUBCASE200} \quad 1.0U_{SUBCASE300}\right]\begin{Bmatrix} x_1 \\ x_2 \end{Bmatrix}$$

Equation 7-11.

Thus, each shape basis vector is simply 1.0 times each of the resultant displacement solutions. (Each solution is that over the entire structure; boundary plus interior grids. The displacement vector subscripts are simply used to indicate the auxiliary boundary model origin of the solution.) Further, a symmetric redesign is enforced by design variable linking with the DLINK entry.

Optimization Results

Figure 7-12 shows the summary of design cycle history from the output file. In five design cycles, the weight has been reduced from 8.0 E+6 to 5.4 E+6. The final shape is shown in Figure 7-13, and the corresponding stress distribution in Figure 7-14. The maximum stress regions, which had been near the root for the initial design, are now located near the tip. This is somewhat expected, since our choice of basis vectors has left the geometry unchanged near the root of the cantilever.

```
                                   ***************************************************************
                                   S U M M A R Y   O F   D E S I G N   C Y C L E   H I S T O R Y
                                   ***************************************************************

                                                      (HARD CONVERGENCE ACHIEVED)

                                                      (SOFT CONVERGENCE ACHIEVED)

                                       NUMBER OF FINITE ELEMENT ANALYSES COMPLETED           6
                                       NUMBER OF OPTIMIZATIONS W.R.T. APPROXIMATE MODELS     5

                                              OBJECTIVE AND MAXIMUM CONSTRAINT HISTORY
           ---------------------------------------------------------------------------------------------------------------
                                OBJECTIVE FROM          OBJECTIVE FROM          FRACTIONAL ERROR          MAXIMUM VALUE
             CYCLE                APPROXIMATE               EXACT                     OF                       OF
             NUMBER              OPTIMIZATION              ANALYSIS              APPROXIMATION             CONSTRAINT
           ---------------------------------------------------------------------------------------------------------------

             INITIAL                                     8.000000E+06                                    -3.827773E-01

                   1             7.401214E+06            7.401214E+06            0.000000E+00            -3.720914E-01

                   2             6.562918E+06            6.562912E+06            9.142283E-07            -3.570341E-01

                   3             5.389287E+06            5.389288E+06           -2.783299E-07            -4.672768E-02

                   4             5.350312E+06            5.350312E+06            0.000000E+00             8.081818E-04

                   5             5.350312E+06            5.350312E+06            0.000000E+00             8.081055E-04
           ---------------------------------------------------------------------------------------------------------------
    AUXILIARY MODEL 1                                                         APRIL 30, 2004 NX NASTRAN   04/30/04    PAGE   226

                                                        DESIGN VARIABLE HISTORY
 ----------------------------------------------------------------------------------------------------------------------------
  INTERNAL |   EXTERNAL   |              |
   DV. ID. |    DV. ID.   |    LABEL     |   INITIAL    :       1       :       2       :       3       :       4       :       5      : --
 ----------------------------------------------------------------------------------------------------------------------------
         1 |            1 |  UPPER       |   1.0000E+00  :   1.4000E+00 :   1.9600E+00  :  2.7440E+00 :   2.7700E+00 :   2.7700E+00 :
         2 |            2 |  LOWER       |   1.0000E+00  :   1.4000E+00 :   1.9600E+00  :  2.7440E+00 :   2.7700E+00 :   2.7700E+00 : --
 ----------------------------------------------------------------------------------------------------------------------------
0*** USER INFORMATION MESSAGE 6464 (DOM12E)
     RUN TERMINATED DUE TO HARD CONVERGENCE TO AN OPTIMUM AT CYCLE NUMBER =        5.
```

Figure 7-12. Summary of Design Cycle History.

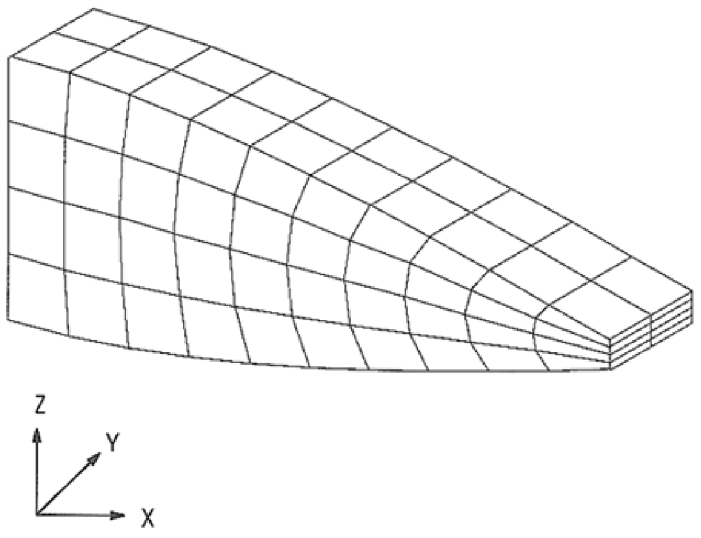


Figure 7-13. Solid Cantilever - Final Stage

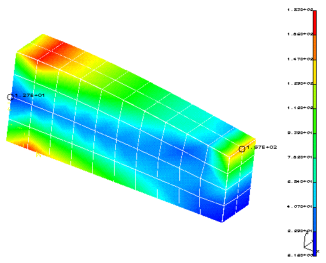


Figure 7-14. Solid Cantilever - Final Stress Distribution

```
TIME 600      $
SOL  200      $
CEND
TITLE = CANTILEVERED BEAM - HEXA **** D200AM3 ****
 DESOBJ = 15
 DESSUB = 100
SUBCASE 100
 ANALYSIS = STATICS
 SPC     = 1
 LOAD    = 1
 DISPLACEMENT = ALL
OUTPUT(PLOT)
SET 1 = ALL
VIEW 90.0,0.0,90.0
PLOT SET 1
$SET 2 = ALL
$VIEW 34.0, 24.0, 0.0
$PLOT SET 2
$PLOT STATIC DEFORMATION SET 2
AUXCASE
 TITLE = AUXILIARY MODEL 1
 AUXMODEL = 1
 SUBCASE 200
  SPC   = 200
  LOAD  = 220
  LABEL = UPPER
 SUBCASE 300
  SPC   = 300
  LOAD  = 330
  LABEL = LOWER
BEGIN BULK
$
$-------------------------------------------------------------------------
$ ANALYSIS MODEL:
$-------------------------------------------------------------------------
$
PARAM   AUTOSPC YES
PARAM   POST    0
PARAM   GRDPNT  0
PARAM   MAXRATIO1.0E+8
$
CORD2S*                2               0             0.0             0.0+1A    2
*1A    2             0.0             0.0             0.0        1000.000+1B    2
*1B    2        1000.000             0.0             0.0                +1C    2
*1C    2
CORD2C*                1               0             0.0             0.0+1A    1
*1A    1             0.0             0.0             0.0        1000.000+1B    1
*1B    1        1000.000             0.0             0.0                +1C    1
*1C    1
GRID           1       0     0.0     0.0     0.0       0
  .            .       .       .       .       .       .
  .            .       .       .       .       .       .
  .            .       .       .       .       .       .
GRID         165       0  10.000   2.000   4.000       0
$GRDSET                                                 456
CHEXA          1       1       1       2      13      12      34      35+EA    1
+EA    1      46      45
  .            .       .       .       .       .       .       .        .
  .            .       .       .       .       .       .       .        .
  .            .       .       .       .       .       .       .        .
CHEXA         80       1     120     121     132     131     153     154+EA   80
+EA   80     165     164
MAT1*                  1      2.0680E+05                     0.28999999166+MA    1
*MA    1      1.00000000 1.169999996E-05                                +MB    1
*MB    1      1500000.00      1500000.00        68000.00                +MC    1
*MC    1
PSOLID         1       1       0       0       0       0
SPC            1       1  123456     0.0
SPC            1      12  123456     0.0
SPC            1      23  123456     0.0
SPC            1      34  123456     0.0
SPC            1      45  123456     0.0
SPC            1      56  123456     0.0
SPC            1      67  123456     0.0
SPC            1      78  123456     0.0
SPC            1      89  123456     0.0
SPC            1     100  123456     0.0
SPC            1     111  123456     0.0
SPC            1     122  123456     0.0
SPC            1     133  123456     0.0
SPC            1     144  123456     0.0                                        SPC            1     155  123456     0.0
SPC1 1 456 1 THRU 165
FORCE          1     143       0     0.5     0.0     0.0   -50.0
FORCE          1     154       0     1.0     0.0     0.0   -50.0
FORCE          1     165       0     0.5     0.0     0.0   -50.0                $
$------------------------------------------------------------------------
$ DESIGN MODEL:
$------------------------------------------------------------------------
$
PARAM,DESPCH,1
PARAM,NASPRT,1
$
$DESVAR,ID,     LABEL,  XINIT,  XLB,    XUB,    DELXV
DESVAR         1 UPPER       1.0     .00    7.00     0.4
DESVAR         2 LOWER       1.0     .00    7.00     0.4
$
$DVBSHAP,DVID,  AUXMID, COL1,   SF1,    COL2,   SF2,    ...
DVBSHAP 1 1 1 1.0
DVBSHAP 2 1 2 1.0
$
$DLINK, ID,     DDVID,  CO,     CMULT,  IDV1,   C1,     IDV2,   C2,     +
$+,     IDV3,   C3,     ...
DLINK 1 2  1.0 1 1.0
$
$ BOUNDARY CONDITIONS FOR SHAPE INTERPOLATIONS:
$
$ ---TOP SURFACE:
$BNDGRID,C,     GP1,    GP2,    GP3,    GP4,    GP5,    GP6,    GP7,    +
$+,     GP8,    ...
BNDGRID 123 133 134 135 136 137 138 139
 140 141 142 143 144 145 146 147
 148 149 150 151 152 153 154 155
 156 157 158 159 160 161 162 163
 164 165
$
$ ---BOTTOM SURFACE:
BNDGRID 123 1 2 3 4 5 6 7
 8 9 10 11 12 13 14 15
 16 17 18 19 20 21 22 23
 24 25 26 27 28 29 30 31
 32 33
$
$ ---EXTERIOR SURFACES - INTERPOLATION IN X&Z DIRECTION ONLY:
BNDGRID 2 34 35 36 37 38 39 40
 41 42 43 44
BNDGRID 2 56 57 58 59 60 61 62
 63 64 65 66
BNDGRID 2 67 68 69 70 71 72 73
 74 75 76 77
```

```
BNDGRID 2 89 90 91 92 93 94 95
 96 97 98 99
BNDGRID 2 100 101 102 103 104 105 106
 107 108 109 110
BNDGRID 2 122 123 124 125 126 127 128
 129 130 131 132
$
$ ---TIP END:
BNDGRID 1 11 22 33 44 55 66
BNDGRID 1 77 88 99 110 121 132
BNDGRID 1 143 154 165
$
$ ---FIXED END:
BNDGRID 123 1 12 23 34 56 67 89
 100 122 133 144 155
BNDGRID 1 45 78 111
$
$ FORMULATE WEIGHT-BASED SYNTHETIC RESPONSE:  F = 1.E5*W
DRESP1  1       WEIGHT  WEIGHT
DRESP2  15      WE1000  1
+       DRESP1  1
DEQATN  1       F(A)=100000.*A
$
$ CONSTRAINTS ON VON MISES STRESSES:
DRESP1  2       STRESS  STRESS  PSOLID                  13                1
DSCREEN STRESS  -1.0    10
DCONSTR 100     2               200.
$
$ OVERRIDE OF OPTIMIZATION PARAMETERS
DOPTPRM DESMAX 9 P1 1 P2 15
$
$------------------------------------------------------------------------
$ AUXILIARY BOUNDARY MODEL(S):
$------------------------------------------------------------------------
$
BEGIN BULK AUXMODEL=1
PARAM,PRGPST,NO
PARAM   MAXRATIO1.0E+8
PARAM,AUTOSPC,YES
$
$ LOWER SURFACE:
$
CQUAD4      1000       2       1       2      13      12     0.0
CQUAD4      1001       2       2       3      14      13     0.0
CQUAD4      1002       2       3       4      15      14     0.0
CQUAD4      1003       2       4       5      16      15     0.0
CQUAD4      1004       2       5       6      17      16     0.0
CQUAD4      1005       2       6       7      18      17     0.0
CQUAD4      1006       2       7       8      19      18     0.0
CQUAD4      1007       2       8       9      20      19     0.0
CQUAD4      1008       2       9      10      21      20     0.0
CQUAD4      1009       2      10      11      22      21     0.0
CQUAD4      1010       2      12      13      24      23     0.0
CQUAD4      1011       2      13      14      25      24     0.0
CQUAD4      1012       2      14      15      26      25     0.0
CQUAD4      1013       2      15      16      27      26     0.0
CQUAD4      1014       2      16      17      28      27     0.0
CQUAD4      1015       2      17      18      29      28     0.0
CQUAD4      1016       2      18      19      30      29     0.0
CQUAD4      1017       2      19      20      31      30     0.0
CQUAD4      1018       2      20      21      32      31     0.0
CQUAD4      1019       2      21      22      33      32     0.0
$
$ UPPER SURFACE:
CQUAD4       950       2     133     144     145     134     0.0
CQUAD4       951       2     134     145     146     135     0.0
CQUAD4       952       2     135     146     147     136     0.0
CQUAD4       953       2     136     147     148     137     0.0
CQUAD4       954       2     137     148     149     138     0.0
CQUAD4       955       2     138     149     150     139     0.0
CQUAD4       956       2     139     150     151     140     0.0
CQUAD4       957       2     140     151     152     141     0.0
CQUAD4       958       2     141     152     153     142     0.0
CQUAD4       959       2     142     153     154     143     0.0
CQUAD4       960       2     144     155     156     145     0.0
CQUAD4       961       2     145     156     157     146     0.0
CQUAD4       962       2     146     157     158     147     0.0
CQUAD4       963       2     147     158     159     148     0.0
CQUAD4       964       2     148     159     160     149     0.0
CQUAD4       965       2     149     160     161     150     0.0
CQUAD4       966       2     150     161     162     151     0.0
CQUAD4       967       2     151     162     163     152     0.0
CQUAD4       968       2     152     163     164     153     0.0
CQUAD4       969       2     153     164     165     154     0.0
$
MAT1          11  2.1E+5  0.8E+5     0.3    0.00
PSHELL         2      11    0.20      11                              0.0
SPC1         200  123456       1      12      23
SPC1 200 12 11 22 33
SPC1 200 123456 34 THRU 165
SPCD 220 11 3 1.0 22 3 1.0
SPCD 220 33 3 1.0
SPC1 200 3 11 22 33
$
SPC1         300  123456     133     144     155                                 SPC1 300 12 143 154 165
SPC1 300 123456 1 THRU 132
SPCD 330 143 3 -1.0 154 3 -1.0
SPCD 330 165 3 -1.0
SPC1 300 3 143 154 165
ENDDATA
```

Listing 7-7.

7.7 Dynamic Response Optimization

This example demonstrates structural optimization when the structural loads are frequency-dependent. The system considered is a flat rectangular plate clamped on three edges and free along the fourth, as shown in Figure 7-15. The problem investigates minimization of the mean square response of the transverse displacement at the midpoint of the free edge, while constraining the volume of the structure (and hence, weight) to be equal to that of the initial design. A pressure loading with an amplitude of 1.0 lb_f/in^2 is applied across a frequency range of 20.0 to 200.0 Hz. A small amount of frequency-dependent modal damping has also been included.

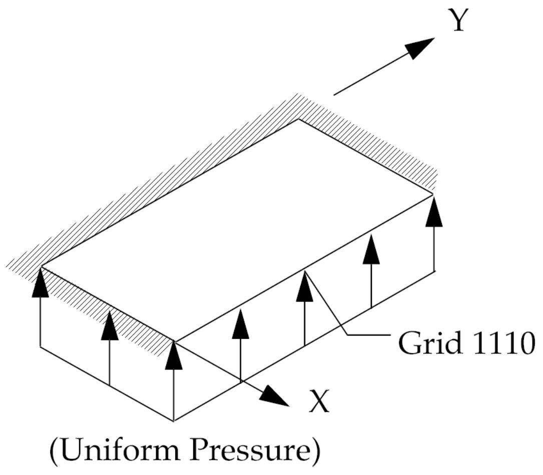


Figure 7-15. Pressure-Loaded Flat Plate

Figure 7-16 shows the finite element representation. Due to symmetry conditions, only half of the structure needs to be modeled. Ten design variables are related to ten plate element property group thicknesses. The first such group is shown in the figure as the shaded "ring" of elements. Subsequent design variables control the thicknesses of subsequent rings of elements up to the tenth design variable that controls the single element connected to GRID1110.

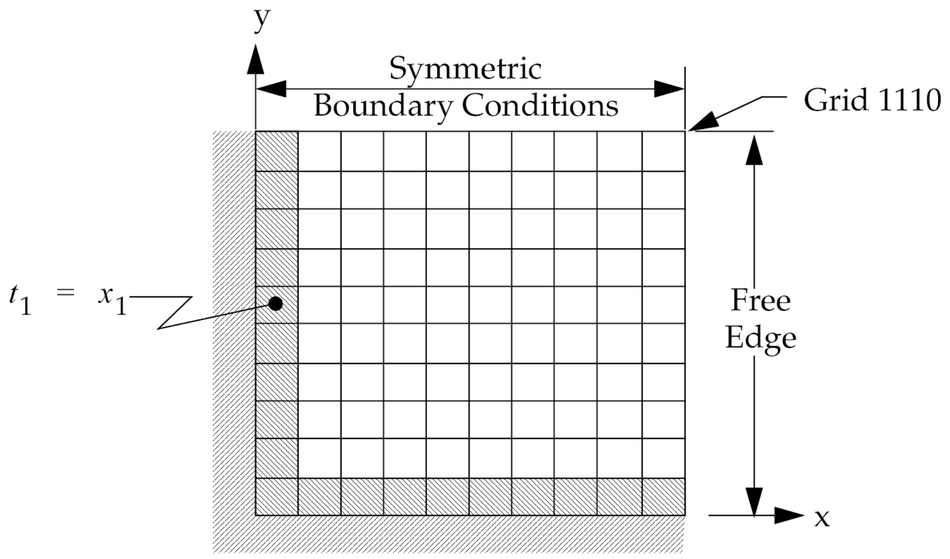


Figure 7-16. Clamped-Free Plate

The dynamic excitation is applied at 180 equal intervals across the range of frequencies. The objective is to minimize the mean sum of the squares of the transverse displacements at GRID 1110, or:

$$min\ \phi = \sum_{i=20}^{100} (u^{i}_{z,1110})^2 + 2\sum_{i=51}^{100} (u^{2i}_{z,1110})^2$$

Equation 7-12.

The above notation indicates that the square of the displacement responses are summed over 1.0 Hz intervals from 20.0 to 100.0 Hz, and over 2.0 Hz intervals from 102.0 to 200.0 Hz. This particular form was selected simply for illustrative purposes; other mean square measures could serve equally as well.

Fifteen modes were retained for the modal representation. This number was selected based on a convergence study which indicated good sensitivity information was obtained with this amount.

Note

A guideline recommendation is to retain at least twice as many modes for modal sensitivity and optimization studies as would be appropriate for an analysis. Of course, the resultant sensitivities should be checked for accuracy and the number increased, if necessary.

From the input file Listing 7-9, it can be seen that the thickness distribution of the initial design is a constant 0.08 inches. The corresponding initial volume is 8.0 in^3. The volume constraint is imposed as follows: the volume response is first identified on DRESP1 number 201, bounds are placed on this response by DCONSTR number 10, which is in turn selected as a global constraint in Case Control (global, since volume is a subcase-independent response) using the DESGLB Case Control command. This results in:

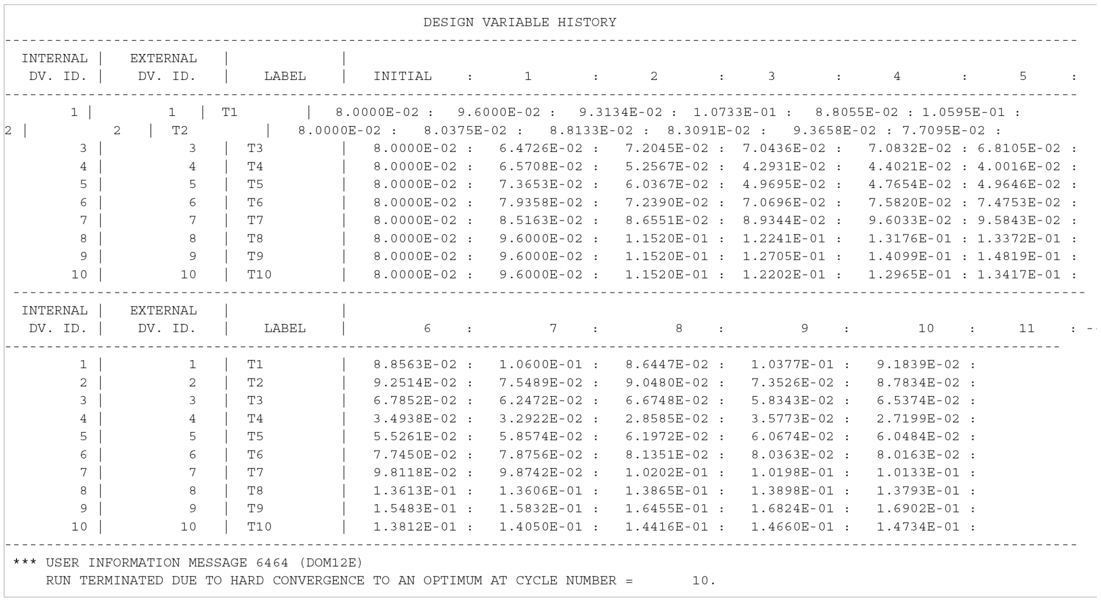
$$7.99 \le vol \le 8.01$$

Equation 7-13.

The mean square response is to be the objective function. The underlying transverse displacements are first identified by DRESP1 entries 20 through 100 and 102 through 200 (note the use of Bulk Data replicators). These identify the transverse component of displacement at GRID 1110 across the frequency range of interest. These first-level responses are then used as input to DEQATN,1 via DRESP2 number 1, which defines the equation input. The resultant response is then defined as the objective function using the Case Control DESOBJ command.

Optimization results are shown in Figure 7-17 and Figure 7-18. Figure 7-17 shows the design cycle history where it can be seen that the objective has been reduced from 230.13 to 133.33 in^2. This took ten design cycles. (Note that CONV1, the relative measure of convergence with respect to the design objective, has been set to 0.01 on the DOPTPRM entry. This indicates convergence as long as the objective is changing less than 1% on subsequent design cycles. Better results were obtained with the default, but it required twenty design cycles to achieve an objective of 128.47 in^2 —not that much better considering the cost involved.)

```
                                  ***************************************************************
                                  S U M M A R Y   O F   D E S I G N   C Y C L E   H I S T O R Y
                                  ***************************************************************

                                                  (HARD CONVERGENCE ACHIEVED)

                                      NUMBER OF FINITE ELEMENT ANALYSES COMPLETED          11
                                      NUMBER OF OPTIMIZATIONS W.R.T. APPROXIMATE MODELS    10

                                             OBJECTIVE AND MAXIMUM CONSTRAINT HISTORY
 ---------------------------------------------------------------------------------------------------------------
                   OBJECTIVE FROM          OBJECTIVE FROM          FRACTIONAL ERROR          MAXIMUM VALUE
  CYCLE             APPROXIMATE                EXACT                      OF                      OF
  NUMBER            OPTIMIZATION              ANALYSIS               APPROXIMATION            CONSTRAINT
 ---------------------------------------------------------------------------------------------------------------

  INITIAL                                   2.301266E+02                                      -1.249123E-03

        1            2.077541E+02           1.857625E+02            1.183856E-01              -1.248111E-03

        2            1.750074E+02           1.608924E+02            8.772969E-02              -6.922793E-06

        3            1.561576E+02           1.551918E+02            6.223301E-03              -4.953377E-06

        4            1.506372E+02           1.472406E+02            2.306841E-02              -2.064902E-05

        5            1.401009E+02           1.426366E+02           -1.777738E-02              -9.372439E-04

        6            1.383175E+02           1.392657E+02           -6.808980E-03              -9.459354E-04

        7            1.342377E+02           1.373367E+02           -2.256553E-02              -9.445066E-04

        8            1.324709E+02           1.350924E+02           -1.940550E-02               3.131290E-04

        9            1.296246E+02           1.331847E+02           -2.673070E-02               3.121765E-04

       10            1.281848E+02           1.333292E+02           -3.858460E-02               3.099144E-04
 ---------------------------------------------------------------------------------------------------------------
```



```
                                    DESIGN VARIABLE HISTORY
-----------------------------------------------------------------------------------------------------
 INTERNAL |   EXTERNAL   |            |
  DV. ID. |    DV. ID.   |    LABEL   |  INITIAL    :     1      :     2      :    3      :     4      :     5     :
-----------------------------------------------------------------------------------------------------
       1 |         1   | T1        |  8.0000E-02 :  9.6000E-02 :  9.3134E-02 : 1.0733E-01 :  8.8055E-02 : 1.0595E-01 :
2 |          2    |  T2         |  8.0000E-02 :  8.0375E-02 :  8.8133E-02 : 8.3091E-02 :  9.3658E-02 : 7.7095E-02 :
        3 |          3   | T3         |  8.0000E-02 :  6.4726E-02 :  7.2045E-02 : 7.0436E-02 :  7.0832E-02 : 6.8105E-02 :
        4 |          4   | T4         |  8.0000E-02 :  6.5708E-02 :  5.2567E-02 : 4.2931E-02 :  4.4021E-02 : 4.0016E-02 :
        5 |          5   | T5         |  8.0000E-02 :  7.3653E-02 :  6.0367E-02 : 4.9695E-02 :  4.7654E-02 : 4.9646E-02 :
        6 |          6   | T6         |  8.0000E-02 :  7.9358E-02 :  7.2390E-02 : 7.0696E-02 :  7.5820E-02 : 7.4753E-02 :
        7 |          7   | T7         |  8.0000E-02 :  8.5163E-02 :  8.6551E-02 : 8.9344E-02 :  9.6033E-02 : 9.5843E-02 :
        8 |          8   | T8         |  8.0000E-02 :  9.6000E-02 :  1.1520E-01 : 1.2241E-01 :  1.3176E-01 : 1.3372E-01 :
        9 |          9   | T9         |  8.0000E-02 :  9.6000E-02 :  1.1520E-01 : 1.2705E-01 :  1.4099E-01 : 1.4819E-01 :
       10 |         10   | T10        |  8.0000E-02 :  9.6000E-02 :  1.1520E-01 : 1.2202E-01 :  1.2965E-01 : 1.3417E-01 :
 -----------------------------------------------------------------------------------------------------
 INTERNAL |   EXTERNAL   |            |
  DV. ID. |    DV. ID.   |    LABEL   |     6     :       7     :       8     :      9     :      10     :     11    : -
-----------------------------------------------------------------------------------------------------
        1 |          1   | T1         |  8.8563E-02 :  1.0600E-01 :  8.6447E-02 :  1.0377E-01 :  9.1839E-02 :
        2 |          2   | T2         |  9.2514E-02 :  7.5489E-02 :  9.0480E-02 :  7.3526E-02 :  8.7834E-02 :
        3 |          3   | T3         |  6.7852E-02 :  6.2472E-02 :  6.6748E-02 :  5.8343E-02 :  6.5374E-02 :
        4 |          4   | T4         |  3.4938E-02 :  3.2922E-02 :  2.8585E-02 :  3.5773E-02 :  2.7199E-02 :
        5 |          5   | T5         |  5.5261E-02 :  5.8574E-02 :  6.1972E-02 :  6.0674E-02 :  6.0484E-02 :
        6 |          6   | T6         |  7.7450E-02 :  7.8756E-02 :  8.1351E-02 :  8.0363E-02 :  8.0163E-02 :
        7 |          7   | T7         |  9.8118E-02 :  9.8742E-02 :  1.0202E-01 :  1.0198E-01 :  1.0133E-01 :
        8 |          8   | T8         |  1.3613E-01 :  1.3606E-01 :  1.3865E-01 :  1.3898E-01 :  1.3793E-01 :
        9 |          9   | T9         |  1.5483E-01 :  1.5832E-01 :  1.6455E-01 :  1.6824E-01 :  1.6902E-01 :
       10 |         10   | T10        |  1.3812E-01 :  1.4050E-01 :  1.4416E-01 :  1.4660E-01 :  1.4734E-01 :
-----------------------------------------------------------------------------------------------------
*** USER INFORMATION MESSAGE 6464 (DOM12E)
    RUN TERMINATED DUE TO HARD CONVERGENCE TO AN OPTIMUM AT CYCLE NUMBER =       10.
```

Figure 7-17. Design Cycle History

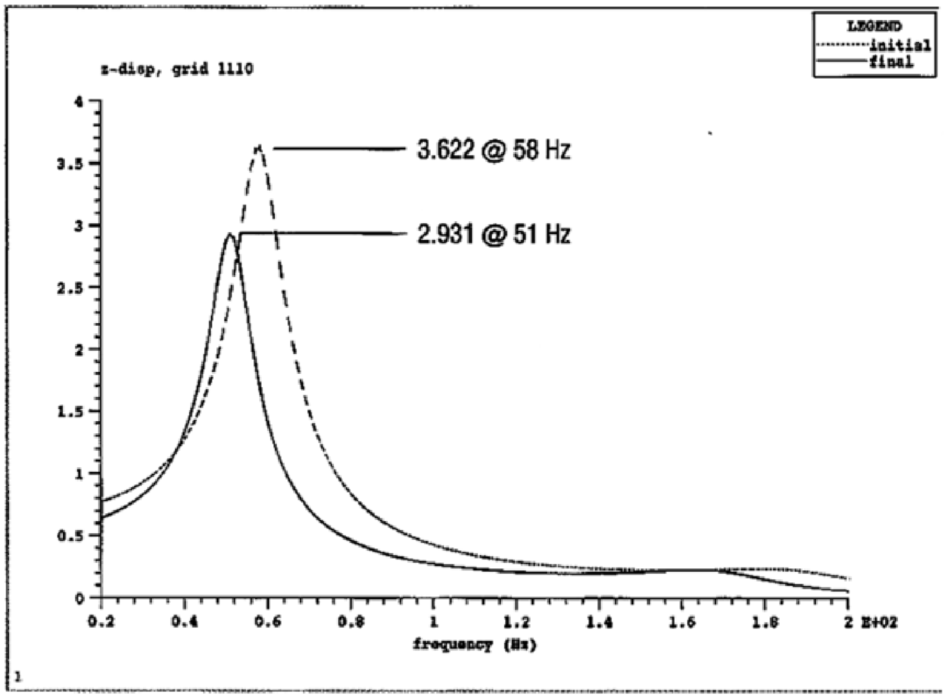


Figure 7-18. Frequency-Dependent Displacements

Figure 7-18 shows the initial and final response data for the displacements used to formulate the objective. Note that the integral of the final curve is indeed less than that of the original curve. In addition, the response peak has shifted somewhat, from a value of 3.622 at 58 Hz to a value of 2.931 at 51 Hz. Figure 7-19 shows the final thickness distribution.

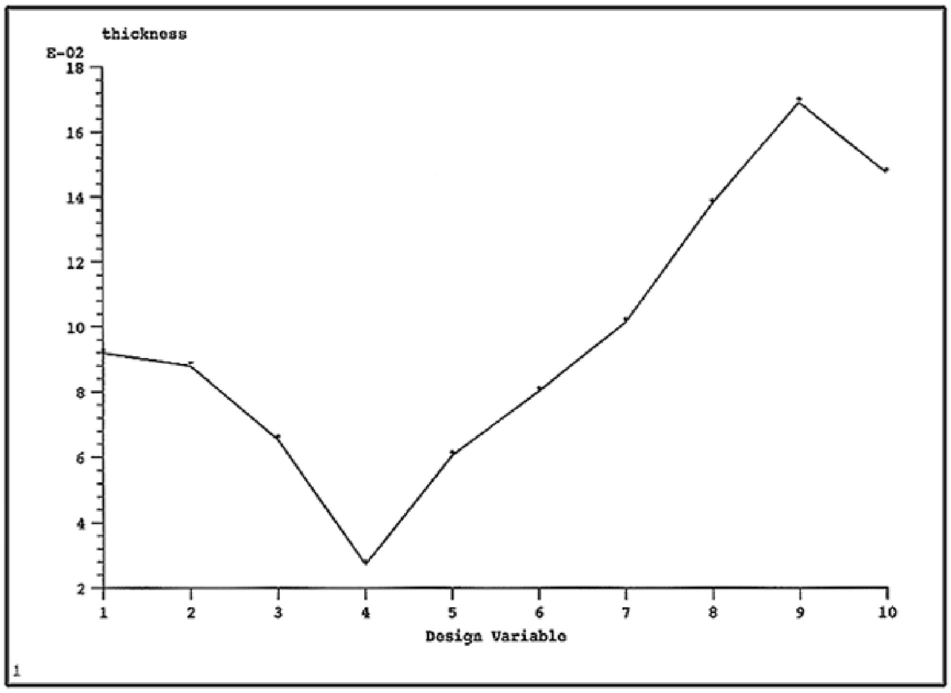


Figure 7-19. Final Thickness Distribution

```
$ Unit Test Data Deck
$    d200t4af: SOL200, V68
$
ID EDS, D200t4af
TIME 100
SOL 200
CEND
TITLE = Synthesis of Responses across Different Frequencies: d200t4af
SET 10 = 1110
DISPL(PHASE,SORT1) = 10   $ MAGNITUDE/PHASE REPRESENTATION FOR RESPONSE
$                           ANALYSIS AS WELL AS SENSITIVITY ANALYSIS
desglb = 10
subcase 1
SPC          = 100
LOADSET      = 720
DLOAD        = 700
FREQ         = 740
METHOD       = 500
ANALYSIS =  MFREQ
sdamping   = 2000
DESOBJ = 1
output
 disp(plot,phase) = 10
output(xyout)
 cscale 2.0
 ymax=4.0
 plotter = nastran
 ytitle = displacement at grid 1110
 xyplot disp / 1110(t3)
$
BEGIN BULK
PARAM, WTMASS, .002588
$-------------------------------------------------------------------------------
$ ANALYSIS MODEL
$-------------------------------------------------------------------------------
$
$...GRID AND SPC DATA:
$
GRDSET, ,       ,       ,       ,       ,       ,       6
GRID,   100,    ,       0.,     0.,     0.
=,      *(1),   ,       *(1.),  =,      =
=9
GRID,   200,    ,       0.,     1.0,    0.
=,      *(1),   ,       *(1.),  =,      =
=9
GRID,   300,    ,       0.,     2.0,    0.
=,      *(1),   ,       *(1.),  =,      =
=9
```

```
GRID,   400,    ,       0.,     3.0,    0.
=,      *(1),   ,       *(1.),  =,      =
=9
GRID,   500,    ,       0.,     4.0,    0.
=,      *(1),   ,       *(1.),  =,      =
=9
GRID,   600,    ,       0.,     5.0,    0.
=,      *(1),   ,       *(1.),  =,      =
=9
GRID,   700,    ,       0.,     6.0,    0.
=,      *(1),   ,       *(1.),  =,      =
=9
GRID,   800,    ,       0.,     7.0,    0.
=,      *(1),   ,       *(1.),  =,      =
=9
GRID,   900,    ,       0.,     8.0,    0.
=,      *(1),   ,       *(1.),  =,      =
=9
GRID,   1000,   ,       0.,     9.0,    0.
=,      *(1),   ,       *(1.),  =,      =
=9
GRID,   1100,   ,       0.,     10.0,   0.
=,      *(1),   ,       *(1.),  =,      =
=9
$
SPC1,   100,    123456  100,    101,    102,    103,    104,    105,    +
+,      106,    107,    108,    109,    110,    200,    300,    400,    +
+,      500,    600,    700,    800,    900,    1000,   1100
SPC1,   100,    246,    1101,   1102,   1103,   1104,   1105,   1106,   +
+,      1107,   1108,   1109
SPC1,   100,    246,    1110
$
$...ELEMENT DEFINITION AND PROPERTIES:
$   (ELEMENTS GROUPED BY PID SINCE THICKNESS OF ALL ELEMENTS IN A GROUP
$    ARE TO BE AFFECTED BY A SINGLE DESIGN VARIABLE)
$
MAT1,   150,    1.0E7,  ,       0.3,    0.1
$...ELEMENT GROUP 1:
CQUAD4, 101,    1,      100,    101,    201,    200
=,      *(100), =,      *(100), *(100), *(100), *(100)
=8
CQUAD4, 102,    1,      101,    102,    202,    201
=,      *(1),   =,      *(1),   *(1),   *(1),   *(1),
=7
PSHELL, 1,      150,    .08,    150
$...ELEMENT GROUP 2:
CQUAD4, 202,    2,      201,    202,    302,    301
=,      *(100), =,      *(100), *(100), *(100), *(100)
=7
CQUAD4, 203,    2,      202,    203,    303,    302
=,      *(1),   =,      *(1),   *(1),   *(1),   *(1),
=6
PSHELL, 2,      150,    .08,    150
$...ELEMENT GROUP 3:
CQUAD4, 303,    3,      302,    303,    403,    402
=,      *(100), =,      *(100), *(100), *(100), *(100)
=6
CQUAD4, 304,    3,      303,    304,    404,    403
=,      *(1),   =,      *(1),   *(1),   *(1),   *(1),
=5
PSHELL, 3,      150,    .08,    150
$...ELEMENT GROUP 4:
CQUAD4, 404,    4,      403,    404,    504,    503
=,      *(100), =,      *(100), *(100), *(100), *(100)
=5
CQUAD4, 405,    4,      404,    405,    505,    504
=,      *(1),   =,      *(1),   *(1),   *(1),   *(1),
=4
PSHELL, 4,      150,    .08,    150
$...ELEMENT GROUP 5:
CQUAD4, 505,    5,      504,    505,    605,    604
=,      *(100), =,      *(100), *(100), *(100), *(100)
=4
CQUAD4, 506,    5,      505,    506,    606,    605
=,      *(1),   =,      *(1),   *(1),   *(1),   *(1),
=3
PSHELL, 5,      150,    .08,    150
$...ELEMENT GROUP 6:
CQUAD4, 606,    6,      605,    606,    706,    705
=,      *(100), =,      *(100), *(100), *(100), *(100)
=3
CQUAD4, 607,    6,      606,    607,    707,    706
=,      *(1),   =,      *(1),   *(1),   *(1),   *(1),
=2
PSHELL, 6,      150,    .08,    150
$...ELEMENT GROUP 7:
CQUAD4, 707,    7,      706,    707,    807,    806
=,      *(100), =,      *(100), *(100), *(100), *(100)
=2
CQUAD4, 708,    7,      707,    708,    808,    807
=,      *(1),   =,      *(1),   *(1),   *(1),   *(1),
=1
PSHELL, 7,      150,    .08,    150
$...ELEMENT GROUP 8:
CQUAD4, 808,    8,      807,    808,    908,    907
=,      *(100), =,      *(100), *(100), *(100), *(100)
=1
CQUAD4, 809,    8,      808,    809,    909,    908
=,      *(1),   =,      *(1),   *(1),   *(1),   *(1),
PSHELL, 8,      150,    .08,    150
$...ELEMENT GROUP 9:
CQUAD4, 909,    9,      908,    909,    1009,   1008
=,      *(100), =,      *(100), *(100), *(100), *(100)
CQUAD4, 910,    9,      909,    910,    1010,   1009
PSHELL, 9,      150,    .08,    150
$...ELEMENT GROUP 10:
CQUAD4, 1010,   10,     1009,   1010,   1110,   1109
PSHELL, 10,     150,    .08,    150
$
$...EIGENVALUE EXTRACTION INFORMATION - 15 RETAINED MODES
$
EIGRL 500   15 0
$
$...FREQUENCY DEPENDENT LOADING DATA: (OSCILLATORY PRESSURE LOAD)
$
RLOAD1, 700,    710,    ,       ,       800
LSEQ,   720,    710,    730
PLOAD2, 730,    1.0,    101,    THRU,   110
PLOAD2, 730,    1.0,    201,    THRU,   210
PLOAD2, 730,    1.0,    301,    THRU,   310
PLOAD2, 730,    1.0,    401,    THRU,   410
PLOAD2, 730,    1.0,    501,    THRU,   510
PLOAD2, 730,    1.0,    601,    THRU,   610
PLOAD2, 730,    1.0,    701,    THRU,   710
PLOAD2, 730,    1.0,    801,    THRU,   810
PLOAD2, 730,    1.0,    901,    THRU,   910
PLOAD2, 730,    1.0,    1001,   THRU,   1010
TABLED1,800,    ,       ,       ,       ,       ,       ,       ,       +
+,      0.0,    1.0,    1.0E3,  1.0,    ENDT
$
FREQ1 740 20. 1. 180
tabdmp1 2000
 0.0 0.20 1000.0 0.20 endt
$
$--------------------------------------------------------------------------------
$ DESIGN MODEL
$--------------------------------------------------------------------------------
$
$...SPECIFY DESIGN VARIABLES, RELATE LINEARLY TO PLATE THICKNESS
DESVAR, 1,      T1,     .08,    .001,    1.0
DESVAR, 2,      T2,     .08,    .001,    1.0
DESVAR, 3,      T3,     .08,    .001,    1.0
DESVAR, 4,      T4,     .08,    .001,    1.0
DESVAR, 5,      T5,     .08,    .001,    1.0
DESVAR, 6,      T6,     .08,    .001,    1.0
DESVAR, 7,      T7,     .08,    .001,    1.0
DESVAR, 8,      T8,     .08,    .001,    1.0
DESVAR, 9,      T9,     .08,    .001,    1.0
DESVAR, 10,     T10,    .08,    .001,    1.0
$
$...RELATE DESIGN VARIABLES TO PLATE THICKNESSES
DVPREL1,101,    PSHELL, 1,      4,      .01,    ,       ,       ,       +00
=,      *(1),   =,      *(1),   =,      =,      =,      =,      =,      *(1)
=8
+00,    1,      1.0
*(1),   *(1),   =
=8
$
DRESP1  201     VOLUME  VOLUME
DRESP1  20  g1110L FRDISP   3 20.0 1110
=       *(1)    =   = = = = *(1.0) =
=79
DRESP1  102  G1110H FRDISP   3 102.0 1110
= *(2) = = = = = *(2.0) =
=48
$
DRESP2 1 UZ2 1
 DRESP1 20 21 22 23 24 25 26
  27 28 29 30 31 32 33
  34 35 36 37 38 39 40
  41 42 43 44 45 46 47
  48 49 50 51 52 53 54
  55 56 57 58 59 60 61
  62 63 64 65 66 67 68
  69 70 71 72 73 74 75
  76 77 78 79 80 81 82
  83 84 85 86 87 88 89
  90 91 92 93 94 95 96
  97 98 99 100 102 104 106
  108 110 112 114 116 118 120
  122 124 126 128 130 132 134
  136 138 140 142 144 146 148
  150 152 154 156 158 160 162
  164 166 168 170 172 174 176
  178 180 182 184 186 188     190
  192 194 196 198 200
$
DEQATN 1 UZ2(U20,u21,u22,u23,u24,u25,U26,U27,U28,U29,U30,
                    U31,U32,U33,U34,U35,U36,U37,U38,U39,U40,
                    U41,U42,U43,U44,U45,U46,U47,U48,U49,U50,
                    U51,U52,U53,U54,U55,U56,U57,U58,U59,U60,
                    U61,U62,U63,U64,U65,U66,U67,U68,U69,U70,
                    U71,U72,U73,U74,U75,U76,U77,U78,U79,U80,
                    U81,U82,U83,U84,U85,U86,U87,U88,U89,U90,
                    U91,U92,U93,U94,U95,U96,U97,U98,U99,U100,
                    U102,U104,U106,U108,U110,U112,U114,U116,U118,U120,
                    U122,U124,U126,U128,U130,U132,U134,U136,U138,U140,
                    U142,U144,U146,U148,U150,U152,U154,U156,U158,U160,
                    U162,U164,U166,U168,U170,U172,U174,U176,U178,U180,
                    U182,U184,U186,U188,U190,U192,U194,U196,U198,U200)
                        = u20**2 + u21**2 + U22**2 + U23**2+ U24**2 +
                 U25**2 + U26**2 + U27**2 + U28**2 + U29**2 +U30**2 +
                 U31**2 + U32**2 + U33**2 + U34**2 + U35**2 +
                 U36**2 + U37**2 + U38**2 + U39**2 + U40**2 +
                 U41**2 + U42**2 + U43**2 + U44**2 + U45**2 +
                 U46**2 + U47**2 + U48**2 + U49**2 + U50**2 +
                 U50**2 + U52**2 + U53**2 + U54**2 + U55**2 +
                 u56**2 + u57**2 + u58**2 + u59**2 + u60**2 +
                 U61**2 + U62**2 + U63**2 + U64**2 + U65**2 +
                 U66**2 + U67**2 + U68**2 + U69**2 + U70**2 +
                 U71**2 + U72**2 + U73**2 + U74**2 + U75**2 +
                 U76**2 + U77**2 + U78**2 + U79**2 + U80**2 +
                 U80**2 + U82**2 + U83**2 + U84**2 + U85**2 +
                 u86**2 + u87**2 + u88**2 + u89**2 + u90**2 +
                 U90**2 + U92**2 + U93**2 + U94**2 + U95**2 +
                 u96**2 + u97**2 + u98**2 + u99**2 + u100**2 +
                 2.0*(u102**2 + u104**2 + u106**2 + u108**2 + u110**2 +
                 u112**2 + u114**2 + u116**2 + u118**2 + u120**2 +
                 u122**2 + u124**2 + u126**2 + u128**2 + u130**2 +
                 u132**2 + u134**2 + u136**2 + u138**2 + u140**2 +
                 u142**2 + u144**2 + u146**2 + u148**2 + u150**2 +
                 u152**2 + u154**2 + u156**2 + u158**2 + u160**2 +
                 u162**2 + u164**2 + u166**2 + u168**2 + u170**2 +
                 u172**2 + u174**2 + u176**2 + u178**2 + u180**2 +
                 u182**2 + u184**2 + u186**2 + u188**2 + u190**2 +
                 u192**2 + u194**2 + u196**2 + u198**2 + u200**2)
$
DCONSTR 10      201     7.99    8.01
doptprm desmax 40 p1 1 p2 8       conv1   0.01
$.......2.......3.......4.......5.......6.......7.......8.......9.......0
$
$param, optexit, 2
ENDDATA
```

Listing 7-8.

7.8 Twenty-Five Bar Truss, Superelement Optimization

This problem, often seen in the design optimization literature, calls for a minimum weight structure subject to member stress, Euler buckling, and joint displacement constraints. The structure is shown in Figure 7-20. The formulation of the buckling constraints is a good example of constructing normalized constraints based on user-defined structural responses.

In addition, this problem will be substructured in order to illustrate superelement optimization.

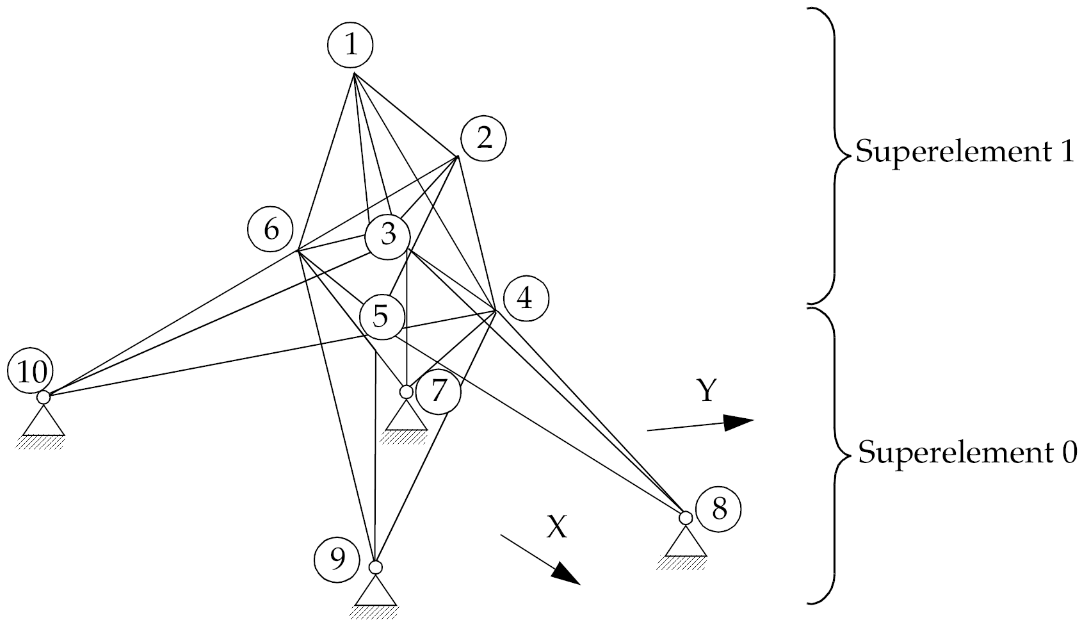


Figure 7-20. Twenty-Five Bar Truss

Analysis Model Description

Three-dimensional truss tower

Symmetric with respect to the x-z plane and y-z plane

Weight density = 0.1 lbs/in^3

Materials: E = 1.0E7 psi

Two distinct loading conditions

Design Model Description

Minimization of structural weight

Design variables:	Cross-sectional areas linked to eight independent design variables
Constraints:	Allowable stress: Tensile = 40,000 psi Compressive = -40,000 psi
	Displacement constraints: +0.35 inches at grid 1 and 2 for all translational degrees of freedom
	Euler buckling constraints for compressive members assuming tubular section diameter- to-thickness ratio of approximately 100

Euler buckling occurs when the magnitude of a member's compressive stress is greater than a critical stress that, for the first buckling mode of a pin-connected member, is

$$\sigma_b = \frac{P_b}{A} = \frac{1}{A}\left(-\frac{\pi^2 EI}{L^2}\right)$$

Equation 7-14.

We will assume thin-walled tubular members with a diameter-to-thickness ratio of approximately 100. Under this condition, the cross-sectional area A is nearly equal to

$$A \cong \pi D t \ , \qquad \text{where } \frac{D}{t} \cong 100$$

Equation 7-15.

and the area moment of inertia can be approximated by

$$I \cong \frac{\pi D t (D^2 + t^2)}{8}$$

Equation 7-16.

Since cross-sectional areas are chosen as design variables, we seek to express the member buckling stresses as functions of member areas as follows:

$$\sigma_b = -\frac{AE}{L^2} \cdot \frac{\pi(100^2 + 1)}{8 \cdot 100} \qquad \text{where } \frac{D}{t} = 100$$

Equation 7-17.

For the buckling stress constraints to be satisfied, the member stresses must not be more compressive than σ_b , that is,

$$\sigma \geq \sigma_b$$

Equation 7-18.

where σ_b is a negative quantity, representing a compressive stress.

Since the constraint in the form of Eq. 7-18 can't be used directly on a DCONSTR entry (the bound is a function of the design variables), we can first normalize the expression as

$$\frac{\sigma}{\sigma_b} \leq 1$$

Equation 7-19.

This is now a well-posed constraint, since the feasible condition has now been normalized to be on the order of 1.0. Since the lower bound is not of interest (e.g. tensile stresses on the elements will not induce Euler buckling), we can just set it to a large negative number on the DCONSTR entry, say -1.0E10.

Eight independent design variables (DESVAR 1 through 8) are used to describe eight ROD cross-sectional areas on DVPREL1 entries 1 through 8. Axial stresses and grid displacements are identified on DRESP1 entries 1 through 14 and are constrained using DCONSTR entries with an I.D. of 10. Additionally, the axial rod stresses are used as input in the definition of the Euler buckling responses (DRESP2s 16 through 23), which all reference DEQATN,1. Note that element lengths are included in these relations via constants defined on the DTABLE entry. This simplifies the input since the element lengths do not need to be hard-coded on eight different DEQATN entries.

Turning to the superelement analysis model, in Case Control the SUPER = ALL command is recommended, rather than an explicit data recovery subcase for each superelement. In Bulk Data, the SESET has been used to place grids 1 and 2 on the interior of superelement 1.

As far as the superelement aspects of the design model are concerned, the only necessary addition is in field 9 are of the DRESP1 number 15 entry. Here, "ALL" has been used to indicate the total structural weight across all superelements be computed and used as the objective Function. The default zero value for this field would have computed the weight using just those elements in the residual structure.

Note that the initial design is infeasible as indicated by the maximum constraint value in the summary of design cycle history. After eight design cycles and a total of nine finite element analyses, a lighter weight structure is found that satisfies all constraints.

```
 ***** NORMAL CONVERGENCE CRITERIA SATISFIED ***** (HARD CONVERGENCE DECISION LOGIC)
*************************************************************************************
                CONVERGENCE ACHIEVED BASED ON THE FOLLOWING CRITERIA
                         (HARD CONVERGENCE DECISION LOGIC)
       RELATIVE CHANGE IN OBJECTIVE        5.2768E-05  MUST BE LESS THAN   1.0000E-03
 OR    ABSOLUTE CHANGE IN OBJECTIVE        2.8809E-02  MUST BE LESS THAN   1.0000E-02
                                  --- AND ---
       MAXIMUM CONSTRAINT VALUE            3.8660E-04  MUST BE LESS THAN   5.0000E-03
                        (CONVERGENCE TO A FEASIBLE DESIGN)
                                  --- OR ---
       MAXIMUM OF RELATIVE PROP. CHANGES   3.8638E-02  MUST BE LESS THAN   1.0000E-03
 AND   MAXIMUM OF RELATIVE D.V. CHANGES    3.8638E-02  MUST BE LESS THAN   1.0000E-03
               (CONVERGENCE TO A BEST COMPROMISE INFEASIBLE DESIGN)
*************************************************************************************


                  ***************************************************************
                  S U M M A R Y   O F   D E S I G N   C Y C L E   H I S T O R Y
                  ***************************************************************
                                   (HARD CONVERGENCE ACHIEVED)
                      NUMBER OF FINITE ELEMENT ANALYSES COMPLETED          9
                      NUMBER OF OPTIMIZATIONS W.R.T. APPROXIMATE MODELS    8
                             OBJECTIVE AND MAXIMUM CONSTRAINT HISTORY
---------------------------------------------------------------------------------------------------------------
                OBJECTIVE FROM           OBJECTIVE FROM          FRACTIONAL ERROR          MAXIMUM VALUE
  CYCLE           APPROXIMATE                 EXACT                    OF                        OF
  NUMBER          OPTIMIZATION               ANALYSIS              APPROXIMATION              CONSTRAINT
---------------------------------------------------------------------------------------------------------------
  INITIAL                                 6.614414E+02                                         1.102773E-01
       1          5.985143E+02            5.985219E+02            -1.254311E-05                2.294609E-03
       2          5.763787E+02            5.763831E+02            -7.730217E-06                8.119856E-04
       3          5.605763E+02            5.605792E+02            -5.117301E-06                3.697191E-04
       4          5.513699E+02            5.513676E+02             4.206515E-06                2.951452E-03
       5          5.484974E+02            5.484994E+02            -3.560852E-06                3.337860E-05
       6          5.465300E+02            5.465295E+02             7.817438E-07                1.285757E-04
       7          5.459542E+02            5.459532E+02             1.788729E-06                3.235681E-06
       8          5.459244E+02            5.459244E+02             0.000000E+00                3.865957E-04
 ---------------------------------------------------------------------------------------------------------------
```

```
DESIGN VARIABLE HISTORY
------------------------------------------------------------------------------------------------------------------------------
  INTERNAL |   EXTERNAL   |              |
   DV. ID. |    DV. ID.   |    LABEL     |   INITIAL    :      1       :      2       :     3       :      4       :      5     :
 -------------------------------------------------------------------------------------------------------------------------------
         1 |         1    |  X1          |   2.0000E+00 :   1.0000E+00 :   9.3000E-01 :  4.6500E-01 :   2.3250E-01 : 1.1625E-01 :
         2 |         2    |  X2          |   2.0000E+00 :   2.0461E+00 :   2.0175E+00 :  2.0560E+00 :   2.0416E+00 : 2.0378E+00 :
         3 |         3    |  X3          |   2.0000E+00 :   3.0000E+00 :   3.1355E+00 :  3.0361E+00 :   3.0550E+00 : 3.0211E+00 :
         4 |         4    |  X4          |   2.0000E+00 :   1.0000E+00 :   8.6000E-01 :  4.3000E-01 :   2.1500E-01 : 1.0750E-01 :
         5 |         5    |  X5          |   2.0000E+00 :   1.0000E+00 :   8.6240E-01 :  4.3120E-01 :   2.1558E-01 : 1.0774E-01 :
         6 |         6    |  X6          |   2.0000E+00 :   1.0000E+00 :   6.1363E-01 :  7.1806E-01 :   6.8686E-01 : 6.6251E-01 :
         7 |         7    |  X7          |   2.0000E+00 :   1.5944E+00 :   1.5890E+00 :  1.5927E+00 :   1.5991E+00 : 1.6201E+00 :
         8 |         8    |  X8          |   2.0000E+00 :   2.5863E+00 :   2.7106E+00 :  2.6171E+00 :   2.6308E+00 : 2.6893E+00 :
------------------------------------------------------------------------------------------------------------------------------
  INTERNAL |   EXTERNAL   |              |
   DV. ID. |    DV. ID.   |    LABEL     |        6     :         7    :         8    :        9    :        10    :        11 :
------------------------------------------------------------------------------------------------------------------------------
         1 |         1    |  X1          |   5.8125E-02 :   2.9063E-02 :   2.8529E-02 :
         2 |         2    |  X2          |   2.0443E+00 :   2.0427E+00 :   2.0427E+00 :
         3 |         3    |  X3          |   3.0237E+00 :   3.0059E+00 :   3.0061E+00 :
         4 |         4    |  X4          |   5.3750E-02 :   2.6875E-02 :   2.5837E-02 :
         5 |         5    |  X5          |   5.3789E-02 :   5.4376E-02 :   5.3374E-02 :
         6 |         6    |  X6          |   6.8365E-01 :   6.8306E-01 :   6.8303E-01 :
         7 |         7    |  X7          |   1.6118E+00 :   1.6216E+00 :   1.6215E+00 :
         8 |         8    |  X8          |   2.6649E+00 :   2.6690E+00 :   2.6691E+00 :
------------------------------------------------------------------------------------------------------------------------------
 *** USER INFORMATION MESSAGE 6464 (DOM12E)
     RUN TERMINATED DUE TO HARD CONVERGENCE TO AN OPTIMUM AT CYCLE NUMBER =        8.
```

```
ID UGS, D200X3
TIME  10
SOL 200       $  OPTIMIZATION
CEND
super = all
ECHO     = UNSORT
OLOAD    = ALL
DISP     = ALL
SPCFORCE = ALL
ELFORCE  = ALL
STRESS   = ALL
SPC      = 100
ANALYSIS = STATICS $
DESOBJ(MIN) = 15      $ OBJECTIVE FUNCTION DEFINITION
DESSUB = 12           $ CONSTRAINT DEFINININITION
SUBCASE 1
   LABEL = LOAD CONDITION 1
   LOAD = 300
SUBCASE 2
   LABEL = LOAD CONDITION 2
   LOAD = 310
BEGIN BULK
$
$-----------------------------------------------------------------------------
$ ANALYSIS MODEL
$-----------------------------------------------------------------------------
$
GRDSET, ,       ,       ,       ,       ,       ,       456
SESET,  1,      1,      2
MAT1,   1,      10.0E6, ,       ,       0.1,    ,       ,       ,       +M1
+M1,    25000., 25000.
SPC1,   100,    123,    7,      THRU,   10
GRID,   1 ,     ,        -37.5,    0.0, 200.0
GRID,   2 ,     ,         37.5,    0.0, 200.0
GRID,   3 ,     ,        -37.5,   37.5, 100.0
GRID,   4 ,     ,         37.5,   37.5, 100.0
GRID,   5 ,     ,         37.5,  -37.5, 100.0
GRID,   6 ,     ,        -37.5,  -37.5, 100.0
GRID,   7 ,     ,       -100.0,  100.0,   0.0
GRID,   8 ,     ,        100.0,  100.0,   0.0
GRID,   9 ,     ,        100.0, -100.0,   0.0
GRID,   10,     ,       -100.0, -100.0,   0.0
CROD,   1 ,     1,      1,      2
CROD,   2 ,     2,      1,      4
CROD,   3 ,     2,      2,      3
CROD,   4 ,     2,      1,      5
CROD,   5 ,     2,      2,      6
CROD,   6 ,     3,      2,      4
CROD,   7 ,     3,      2,      5
CROD,   8 ,     3,      1,      3
CROD,   9 ,     3,      1,      6
CROD,   10,     4,      3,      6
CROD,   11,     4,      4,      5
CROD,   12,     5,      3,      4
CROD,   13,     5,      5,      6
CROD,   14,     6,      3,      10
CROD,   15,     6,      6,      7
CROD,   16,     6,      4,      9
CROD,   17,     6,      5,      8
CROD,   18,     7,      4,      7
CROD,   19,     7,      3,      8
```

```
CROD,   20,     7,      5,      10
CROD,   21,     7,      6,      9
CROD,   22,     8,      6,      10
CROD,   23,     8,      3,      7
CROD,   24,     8,      5,      9
CROD,   25,     8,      4,      8
$
PROD,   1,      1,      2.0,    0.0
PROD,   2,      1,      2.0,    0.0
PROD,   3,      1,      2.0,    0.0
PROD,   4,      1,      2.0,    0.0
PROD,   5,      1,      2.0,    0.0
PROD,   6,      1,      2.0,    0.0
PROD,   7,      1,      2.0,    0.0
PROD,   8,      1,      2.0,    0.0
$
FORCE,  300,    1,      ,       1.0,    1000.,   10000.,-5000.
FORCE,  300,    2,      ,       1.0,       0.,   10000.,-5000.
FORCE,  300,    3,      ,       1.0,     500.,       0.,    0.
FORCE,  300,    6,      ,       1.0,     500.,       0.,    0.
FORCE,  310,    1,      ,       1.0,       0.,   20000.,-5000.
FORCE,  310,    2,      ,       1.0,       0.,  -20000.,-5000.
$
$-------------------------------------------------------------------------------
$ DESIGN MODEL
$-------------------------------------------------------------------------------
$
$...Define the design variables
$DESVAR,ID,     LABEL,  XINIT,  XLB,    XUB,    DELXV
$
DESVAR, 1,      X1,     2.0,    0.01,   100.0
DESVAR, 2,      X2,     2.0,    0.01,   100.0
DESVAR, 3,      X3,     2.0,    0.01,   100.0
DESVAR, 4,      X4,     2.0,    0.01,   100.0
DESVAR, 5,      X5,     2.0,    0.01,   100.0
DESVAR, 6,      X6,     2.0,    0.01,   100.0
DESVAR, 7,      X7,     2.0,    0.01,   100.0
DESVAR, 8,      X8,     2.0,    0.01,   100.0
$
$...Relate the design variables to the analysis model properties
$DVPREL1,ID,    TYPE,   PID,    FID,    PMIN,   PMAX,   C0,     ,       +
$+,     DVIDD1, COEF1,  DVID2,  COEF2,  ...
$
DVPREL1,1,      PROD,   1,      4,      ,       ,       ,       ,       +DP1
+DP1   ,1,      1.0
DVPREL1,2,      PROD,   2,      4,      ,       ,       ,       ,       +DP2
+DP2   ,2,      1.0
DVPREL1,3,      PROD,   3,      4,      ,       ,       ,       ,       +DP3
+DP3   ,3,      1.0
DVPREL1,4,      PROD,   4,      4,      ,       ,       ,       ,       +DP4
+DP4   ,4,      1.0
DVPREL1,5,      PROD,   5,      4,      ,       ,       ,       ,       +DP5
+DP5   ,5,      1.0
DVPREL1,6,      PROD,   6,      4,      ,       ,       ,       ,       +DP6
+DP6   ,6,      1.0
DVPREL1,7,      PROD,   7,      4,      ,       ,       ,       ,       +DP7
+DP7   ,7,      1.0
DVPREL1,8,      PROD,   8,      4,      ,       ,       ,       ,       +DP8
+DP8   ,8,      1.0
$
$...Identify the responses to be used in the design model
$DRESP1,ID,     LABEL,  RTYPE,  PTYPE,  REGION, ATTA,   ATTB,   ATT1,   +
$+,     ATT2,   ...
$
DRESP1, 1 ,     S1,     STRESS, PROD,   ,       2,      ,       1
DRESP1, 2 ,     S2,     STRESS, PROD,   ,       2,      ,       2
DRESP1, 3 ,     S3,     STRESS, PROD,   ,       2,      ,       3
DRESP1, 4 ,     S4,     STRESS, PROD,   ,       2,      ,       4
DRESP1, 5 ,     S5,     STRESS, PROD,   ,       2,      ,       5
DRESP1, 6 ,     S6,     STRESS, PROD,   ,       2,      ,       6
DRESP1, 7 ,     S7,     STRESS, PROD,   ,       2,      ,       7
DRESP1, 8 ,     S8,     STRESS, PROD,   ,       2,      ,       8
DRESP1, 9 ,     D1,     DISP  , ,       ,       1,      ,       1
DRESP1, 10,     D2,     DISP  , ,       ,       2,      ,       1
DRESP1, 11,     D3,     DISP  , ,       ,       3,      ,       1
DRESP1, 12,     D4,     DISP  , ,       ,       1,      ,       2
DRESP1, 13,     D5,     DISP  , ,       ,       2,      ,       2
DRESP1, 14,     D6,     DISP  , ,       ,       3,      ,       2
DRESP1, 15,     W ,     WEIGHT, ,       ,       ,       ,       ALL
$
$...Formulate the second level responses (here, simple Euler buckling)
$DRESP2,ID,     LABEL,  EQID,   REGION, ,       ,       ,       ,       +
$+,     DESVAR, DVID1,  DVID2,  ...,    ,       ,       ,       ,       +
$+,     DTABLE, LABEL1, LABEL2, ...,    ,       ,       ,       ,       +
$+,     DRESP1, NR1,    NR2,    ...,    ,       ,       ,       ,       +
$+,     DNODE,  NID1,   DIR1,   NID2,   DIR2,   ...,    ,       ,       +
$
DRESP2, 16,     SC1,    1,      ,       ,       ,       ,       ,       +DR11
+DR11  , DESVAR, 1,     ,       ,       ,       ,       ,       ,       +DR12
+DR12  , DTABLE, L1,    ,       ,       ,       ,       ,       ,       +DR13
+DR13  , DRESP1, 1
DRESP2, 17,     SC2,    1,      ,       ,       ,       ,       ,       +DR21
```

```
+DR21 , DESVAR, 2,      ,       ,       ,       ,       ,       +DR22
+DR22 , DTABLE, L2,     ,       ,       ,       ,       ,       +DR23
+DR23 , DRESP1, 2
DRESP2, 18,     SC3,    1,      ,       ,       ,       ,       +DR31
+DR31 , DESVAR, 3,      ,       ,       ,       ,       ,       +DR32
+DR32 , DTABLE, L3,     ,       ,       ,       ,       ,       +DR33
+DR33 , DRESP1, 3
DRESP2, 19,     SC4,    1,      ,       ,       ,       ,       +DR41
+DR41 , DESVAR, 4,      ,       ,       ,       ,       ,       +DR42
+DR42 , DTABLE, L4,     ,       ,       ,       ,       ,       +DR43
+DR43 , DRESP1, 4
DRESP2, 20,     SC5,    1,      ,       ,       ,       ,       +DR51
+DR51 , DESVAR, 5,      ,       ,       ,       ,       ,       +DR52
+DR52 , DTABLE, L5,     ,       ,       ,       ,       ,       +DR53
+DR53 , DRESP1, 5
DRESP2, 21,     SC6,    1,      ,       ,       ,       ,       +DR61
+DR61 , DESVAR, 6,      ,       ,       ,       ,       ,       +DR62
+DR62 , DTABLE, L6,     ,       ,       ,       ,       ,       +DR63
+DR63 , DRESP1, 6
DRESP2, 22,     SC7,    1,      ,       ,       ,       ,       +DR71
+DR71 , DESVAR, 7,      ,       ,       ,       ,       ,       +DR72
+DR72 , DTABLE, L7,     ,       ,       ,       ,       ,       +DR73
+DR73 , DRESP1, 7
DRESP2, 23,     SC8,    1,      ,       ,       ,       ,       +DR81
+DR81 , DESVAR, 8,      ,       ,       ,       ,       ,       +DR82
+DR82 , DTABLE, L8,     ,       ,       ,       ,       ,       +DR83
+DR83 , DRESP1, 8
$
$...Equations used to define second level responses (note: fixed-field form!!)
$DEQATN EQUID   F() = ...
$
DEQATN  1       F(A,RL,S) = -S*RL**2/(A*1.E7*39.274)
$
$...Table constants
$DTABLE,LABEL1, VALUE1, LABEL2, VALUE2, LABEL3, VALUE3, LABEL4, VALUE4  +
$+,     LABEL5, VALUE5, ...
$
DTABLE, L1,     75.00,  L2,     130.50, L3,     106.80,       L4,      75.00, +
+,      L5,     75.00,  L6,     181.14,      L7,     181.14, L8,     133.46
$
$...Define the design constraints
$CONSTR,DCID,   RID,    LALLOW, UALLOW
$
DCONSTR,10,     1 ,     -40000.,40000.
DCONSTR,10,     2 ,     -40000.,40000.
DCONSTR,10,     3 ,     -40000.,40000.
DCONSTR,10,     4 ,     -40000.,40000.
DCONSTR,10,     5 ,     -40000.,40000.
DCONSTR,10,     6 ,     -40000.,40000.
DCONSTR,10,     7 ,     -40000.,40000.
DCONSTR,10,     8 ,     -40000.,40000.
DCONSTR,10,     9 ,     -0.35  ,0.35
DCONSTR,10,     10,     -0.35  ,0.35
DCONSTR,10,     11,     -0.35  ,0.35
DCONSTR,10,     12,     -0.35  ,0.35
DCONSTR,10,     13,     -0.35  ,0.35
DCONSTR,10,     14,     -0.35  ,0.35
$
DCONSTR,11,     16,     -1.0E10, 1.0
DCONSTR,11,     17,     -1.0E10, 1.0
DCONSTR,11,     18,     -1.0E10, 1.0
DCONSTR,11,     19,     -1.0E10, 1.0
DCONSTR,11,     20,     -1.0E10, 1.0
DCONSTR,11,     21,     -1.0E10, 1.0
DCONSTR,11,     22,     -1.0E10, 1.0
DCONSTR,11,     23,     -1.0E10, 1.0
$
$...Combine the two constraint sets
$   (equivalent to just putting all into the same set to begin with)
DCONADD,12,     10,     11
$
$...Override optimization parameter defaults:
$
DOPTPRM,IPRINT, 3,      DESMAX, 15,     DELP,   0.5,    p1,     1,      +
+,      p2,     15
ENDDATA
$.......2.......3.......4.......5.......6.......7.......8.......9.......0
```

Listing 7-9.

7.9 Design Optimization with Composite Materials

In the optimal design of laminated composite structures, individual ply thicknesses and orientations are often selected as design quantities. This example shows how a simple composites design problem may be solved using NX Nastran.

Figure 7-21 shows a composite tube that is to be subjected to an internal pressure load of 100 psi. The Hill failure theory will be used to ensure a satisfactory design, where an index value in excess of 0.9 will be used to indicate failure. Both ply thicknesses and orientations are to be modified.

The tube is modeled with 160 CQUAD4 elements oriented such that each element x axis is in the hoop direction and the surface normal faces outward. Ply angles are thus measured from the hoop direction, corresponding to 0°. Due to input deck size, the file has been abbreviated in Listing 7-10.

From the PCOMP entry, note that the structure is built up in eight plies. The outer layer pairs are initially at ±85° orientations to the hoop direction, while the four core layers are at ±60° orientations. All plies are of equal thickness and material type.

We would like to vary the overall thickness by allowing the individual plies to change uniformly. The design variable-to-ply thickness relations are specified on DVPREL1 entries 1 through 8 that define

$$t_i = 1.0x_1 \quad ; \quad i = 1, \ldots, 8$$

Equation 7-20.

Thus, all thicknesses are functions of design variable number 1 defined on the DESVAR,1 entry. The minimum allowable thickness is defined by the default value of PMIN (0.001) on the DVPREL1 entry. This also agrees with the design variable lower bound supplied on the DESVAR entry.

Changing the individual ply thickness will cause the overall plate thickness to vary. Z_0, or the distance from the reference plane to the bottom surface must also be changed accordingly. The relation supplied on DVPREL1,100 ensures this by defining

$$Z_0(x_1) = -\frac{1}{2} \cdot 8t_i(x_1)$$

Equation 7-21.

where t_i is given by Eq. 7-20.

Ply orientations are modified using DVPREL1 entries 11 through 18, which express the ply angles as functions of design variables 2 and 3. Negative values of PMIN have been used in DVPREL1 entries 12, 14, 16, and 18, to allow for ply angles that are less than zero.

The design constraints on ply failure are given by the DCONSTR entry set 20 that references the ply failure responses defined on DRESP1 entries 1 through 8. This DCONSTR entry set is then identified by the DESSUB in the Case Control section. This constrains the failure modes for all plies in element 64. Including other element stress constraints was unnecessary due to the symmetry of the structure and the implicit linking of the design parameters. The "CFAILURE" in field 4 of each of the DRESP1 entries indicate that the responses of interest are failure criteria.

The particular failure mode is indicated in part by the information appearing in field 7 (the ATTA field). According to the DRESP1 entry listing, this attribute field contains the "failure criterion item code."

These, as well as the item codes for all other element types are listed in the *NX Nastran Quick Reference Guide*. See Item Codes for more information. Consulting this section for the CQUAD4 element shows that the only applicable choices are 5 for ply failure or 7 for bonding material failure. The ply failure index is determined based on the computed lamina stresses and/or strains, and the particular failure theory selected on the PCOMP entry. With HILL selected on PCOMP,1, the 5 in field 7 of each DRESP1 entry identifies Hill failure indices for use in the Design Model.

Hard convergence was achieved after thirteen design cycles as seen in Figure 7-22 and Figure 7-23. The results indicate that use of minimum allowable ply thicknesses is feasible if corresponding changes in ply orientation are made. The resulting orientations are 54.8° and 9.5° for the outer and inner plies, respectively.

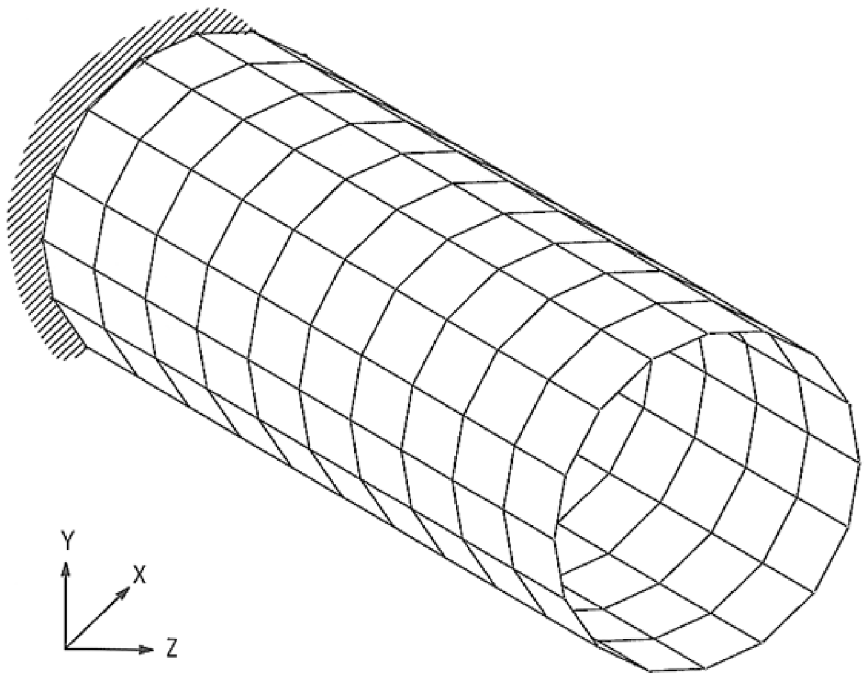


Figure 7-21. Composite Tube

```
***** NORMAL CONVERGENCE CRITERIA SATISFIED ***** (HARD CONVERGENCE DECISION LOGIC)
*************************************************************************************
                 CONVERGENCE ACHIEVED BASED ON THE FOLLOWING CRITERIA
                          (HARD CONVERGENCE DECISION LOGIC)
       RELATIVE CHANGE IN OBJECTIVE        0.0000E+00  MUST BE LESS THAN   1.0000E-03
 OR    ABSOLUTE CHANGE IN OBJECTIVE        0.0000E+00  MUST BE LESS THAN   1.0000E-03
                                   --- AND ---
       MAXIMUM CONSTRAINT VALUE           -1.3644E-01  MUST BE LESS THAN   5.0000E-03
                         (CONVERGENCE TO A FEASIBLE DESIGN)
                                   --- OR ---
       MAXIMUM OF RELATIVE PROP. CHANGES   0.0000E+00  MUST BE LESS THAN   1.0000E-03
 AND   MAXIMUM OF RELATIVE D.V. CHANGES    0.0000E+00  MUST BE LESS THAN   1.0000E-03
                (CONVERGENCE TO A BEST COMPROMISE INFEASIBLE DESIGN)
*************************************************************************************
```

Figure 7-22. Hard Convergence Output for the Composite Tube

```
                                   ***************************************************************
                                   S U M M A R Y   O F   D E S I G N   C Y C L E   H I S T O R Y
                                   ***************************************************************

                                                     (HARD CONVERGENCE ACHIEVED)

                                                     (SOFT CONVERGENCE ACHIEVED)

                                       NUMBER OF FINITE ELEMENT ANALYSES COMPLETED          14
                                       NUMBER OF OPTIMIZATIONS W.R.T. APPROXIMATE MODELS    13

                                              OBJECTIVE AND MAXIMUM CONSTRAINT HISTORY
           ---------------------------------------------------------------------------------------------------------------
                          OBJECTIVE FROM           OBJECTIVE FROM           FRACTIONAL ERROR          MAXIMUM VALUE
             CYCLE          APPROXIMATE                EXACT                      OF                       OF
             NUMBER        OPTIMIZATION               ANALYSIS               APPROXIMATION             CONSTRAINT
           ---------------------------------------------------------------------------------------------------------------

             INITIAL                                1.598129E+00                                        5.114175E-01

                   1          1.118684E+00          1.118690E+00           -5.114954E-06            -8.037044E-02

                   2          7.193502E-01          7.191593E-01            2.653850E-04             1.896803E-01

                   3          5.404640E-01          5.403725E-01            1.694252E-04             1.283122E-01

                   4          4.362053E-01          4.362369E-01           -7.255247E-05             8.706033E-02

                   5          3.744896E-01          3.744358E-01            1.438239E-04             5.622089E-02

                   6          3.359877E-01          3.359930E-01           -1.596586E-05             3.742423E-02

                   7          3.119118E-01          3.119270E-01           -4.891782E-05             2.289958E-02

                   8          2.961263E-01          2.961254E-01            3.119868E-06             1.413279E-02

                   9          2.855329E-01          2.855373E-01           -1.534280E-05             8.562075E-03

                  10          2.767122E-01          2.767129E-01           -2.692531E-06            -9.941631E-02

                  11          2.260737E-01          2.260700E-01            1.634666E-05            -9.497630E-02

                  12          1.608659E-01          1.608551E-01            6.771779E-05            -1.364352E-01

                  13          1.608551E-01          1.608551E-01            0.000000E+00            -1.364352E-01
           ---------------------------------------------------------------------------------------------------------------

                                                      DESIGN VARIABLE HISTORY -----------------------------------------------------
--------------------------------------------------------------------------
  INTERNAL |   EXTERNAL  |             |
   DV. ID. |    DV. ID.  |    LABEL    |   INITIAL    :      1       :      2       :      3      :      4       :      5     :
------------------------------------------------------------------------------------------------------------------------------------
        1 |          1  |  TPLY       |   1.0000E+00 :   7.0000E-01 :   4.5000E-01 :   3.3813E-01 :  2.7297E-01 : 2.3430E-01 :
        2 |          2  |  THETA      |   8.5000E+01 :   8.6324E+01 :   8.6703E+01 :   9.0000E+01 :  9.0000E+01 : 8.9947E+01 :
        3 |          3  |  THETA      |   6.0000E+01 :   4.7320E+01 :   4.1473E+01 :   3.5252E+01 :  2.9964E+01 : 2.5470E+01 :
 ------------------------------------------------------------------------------------------------------------------------------------
  INTERNAL |   EXTERNAL  |             |
   DV. ID. |    DV. ID.  |    LABEL    |        6     :        7     :        8     :        9    :       10     :       11   : ----
------------------------------------------------------------------------------------------------------------------------------
        1 |          1  |  TPLY       |   2.1024E-01 :   1.9518E-01 :   1.8529E-01 :  1.7867E-01 :  1.7315E-01 :  1.4146E-01 :
        2 |          2  |  THETA      |   8.9947E+01 :   8.9825E+01 :   8.9708E+01 :   8.9229E+01 : 7.5845E+01 :  6.4468E+01 :
        3 |          3  |  THETA      |   2.1649E+01 :   1.8402E+01 :   1.5642E+01 :   1.3295E+01 : 1.1301E+01 :  9.6059E+00 :
 ------------------------------------------------------------------------------------------------------------------------------------
  INTERNAL |   EXTERNAL  |             |
   DV. ID. |    DV. ID.  |    LABEL    |       12     :       13     :       14     :       15     :       16     :      17    :
 ------------------------------------------------------------------------------------------------------------------------------------
        1 |          1  |  TPLY       |   1.0065E-01 :   1.0065E-01 :
        2 |          2  |  THETA      |   5.4798E+01 :   5.4798E+01 :
        3 |          3  |  THETA      |   9.5154E+00 :   9.5154E+00 :
 ------------------------------------------------------------------------------------------------------------------------------------
0*** USER INFORMATION MESSAGE 6464 (DOM12E)
     RUN TERMINATED DUE TO HARD CONVERGENCE TO AN OPTIMUM AT CYCLE NUMBER =       13.
```

Figure 7-23. Design History for the Composite Tube

```
ID COMPOSITE,TUBE
SOL 200
TIME 200
CEND
$ ANALYSIS CASE CONTROL:
TITLE    = TUBE UNDER INTERNAL PRESSURE - DIRECT APPROXIMATION
SUBTITLE = ANTI SYMMETRIC ANGLE-PLY [85/-85/60/-60/60/-60/85/-85]
SPC      = 1
LOAD     = 1
SET 1    = 32,64
STRESS   = 1
FORCE    = 1
$ OPTIMIZATION CASE CONTROL:
ANALYSIS    = STATICS
DESOBJ(MIN) = 10
DESSUB      = 20
BEGIN BULK
$
GRID    1       0       -4.     0.0     0.0
GRID    2       0       -3.71915-1.472380.0
GRID    3       0       -2.82843-2.828430.0
  .     .       .          .       .    .
  .     .       .          .       .    .
  .     .       .          .       .    .
GRID    217     0       -1.472383.71915 20.
GRID    218     0       -2.828432.82843 20.
GRID    219     0       -3.719151.47238 20.
$
$
CQUAD4  1       1       1       2       7       6
CQUAD4  2       1       2       3       8       7
CQUAD4  3       1       3       4       9       8
  .     .       .       .       .       .       .
  .     .       .       .       .       .       .
  .     .       .       .       .       .       .
CQUAD4  158     1       212     213     218     217
CQUAD4  159     1       213     214     219     218
CQUAD4  160     1       214     46      51      219
$
$ LAMINATE DEFINITION - ANTISYMMETRIC ANGLE PLY
$ ALL PLIES ARE OF THE SAME MATERIAL KFRP (KEV 49 EPOXY)
$ AND THICKNESS
$
$ PID ZO NSM SB FT TREF GE LAM
PCOMP 1   1.3E4 HILL    +PC1
$ MID1 T1 THETA1 SOUT1 MID2 T2 THETA2 SOUT2
+PC1 100 0.01 85.0 YES   -85.0 YES +PC2
+PC2   60.0 YES   -60.0 YES +PC3
+PC3   60.0 YES   -60.0 YES +PC4
+PC4   85.0 YES   -85.0   YES
$
$ PLY MATERIAL DATA
$
$  ALL ALLOWABLES MUST BE DEFINED AS POSITIVE
$ MID E1 E2 NU12 G12 G1Z G2Z RHO
MAT8 100 1.0701E75.4375E50.4   2.523E52.523E52.523E5   0.04 +M1
$ A1 A2 TREF XT XC YT YC S
+M1    2.426E5 3.3205E43.0305E31.9864E41.2025E4
$ GE F12 STRN$
$ THIS SECTION CONTAINS THE LOADS,CONSTRAINTS, AND CONTROL BULK DATA ENTRIES
$
$
$
PLOAD4  1       1       100.                                            +
+       0
PLOAD4  1       2       100.                                            +
+       0
PLOAD4  1       3       100.                                            +
+       0
  .     .       .       .                                               .
  .     .       .       .                                               .
  .     .       .       .                                               .
PLOAD4  1       158     100.                                            +
+       0
PLOAD4  1       159     100.                                            +
+       0
PLOAD4  1       160     100.                                            +
+       0
$
SPC     1       1       123456  0.0
SPC     1       2       123456  0.0
SPC     1       3       123456  0.0
SPC     1       4       123456  0.0
SPC     1       5       123456  0.0
SPC     1       57      123456  0.0
SPC     1       58      123456  0.0
SPC     1       59      123456  0.0
SPC     1       60      123456  0.0
SPC     1       112     123456  0.0
SPC     1       113     123456  0.0
SPC     1       114     123456  0.0
SPC     1       115     123456  0.0
SPC     1       167     123456  0.0
SPC     1       168     123456  0.0
SPC     1       169     123456  0.0
$
PARAM   AUTOSPC YES
$
$ DESIGN MODEL
$
$ESVAR, ID,     LABEL,  XINIT,  XLB,    XUB,    DELXV
DESVAR 1 TPLY 1.0 0.001 10.0
DESVAR 2 THETA 85.0 -90.0 90.0
DESVAR 3 THETA 60.0 -90.0 90.0
$
$   DV1  - PLY THICKNESS
$VPREL1,ID,     TYPE,   PID,    FID,    PMIN,   PMAX,   C0,     ,      +
$+,     DVID1,  COEF1,  DVID2,  COEF2,  ...
DVPREL1 1 PCOMP 1 13     +DV1
+DV1 1 0.01
DVPREL1 2 PCOMP 1 17     +DV2
+DV2 1 0.01
DVPREL1 3 PCOMP 1 23     +DV3
+DV3 1 0.01
DVPREL1 4 PCOMP 1 27     +DV4
+DV4 1 0.01
DVPREL1 5 PCOMP 1 33     +DV5
+DV5 1 0.01
DVPREL1 6 PCOMP 1 37     +DV6
+DV6 1 0.01
DVPREL1 7 PCOMP 1 43     +DV7
+DV7 1 0.01
DVPREL1 8 PCOMP 1 47     +DV8
+DV8 1 0.01
$
$  DV 2   THETA 85/-85
$  DV 3   THETA 60/-60
$
$VPREL1,ID,     TYPE,   PID,    FID,    PMIN,   PMAX,   C0,     ,       +
$+,     DVID1,  COEF1,  DVID2,  COEF2,  ...
DVPREL1 11 PCOMP 1 14     +DT1
+DT1 2 1.0
DVPREL1 12 PCOMP 1 18 -90.0    +DT2
+DT2 2 -1.0
DVPREL1 13 PCOMP 1 24     +DT3
+DT3 3 1.0
DVPREL1 14 PCOMP 1 28 -90.0    +DT4
+DT4 3 -1.0
DVPREL1 15 PCOMP 1 34     +DT5
+DT5 3 1.0
DVPREL1 16 PCOMP 1 38 -90.0    +DT6
+DT6 3 -1.0
DVPREL1 17 PCOMP 1 44     +DT7
+DT7 2 1.0
DVPREL1 18 PCOMP 1 48 -90.0    +DT8
+DT8 2 -1.0
$
DVPREL1 100 PCOMP 1 3 -100.    +DZ1
+DZ1 1 -0.04
$
$RESP1, ID,     LABEL,  RTYPE,  PTYPE,  REGION, ATTA,   ATTB,   ATT1,   +
$+,     ATT2,   ...
DRESP1 10 W WEIGHT
$DESOBJ 10 W  MIN
DRESP1 1 FP CFAILUREELEM   5 1 64
DRESP1 2 FP CFAILUREELEM   5 2 64
DRESP1 3 FP CFAILUREELEM   5 3 64
DRESP1 4 FP CFAILUREELEM   5 4 64
DRESP1 5 FP CFAILUREELEM   5 5 64
DRESP1 6 FP CFAILUREELEM   5 6 64
DRESP1 7 FP CFAILUREELEM   5 7 64
DRESP1 8 FP CFAILUREELEM   5 8 64
$
$CONSTR,DCID,   RID,    LALLOW, UALLOW
DCONSTR,20,     1,      .001,   0.9
DCONSTR,20,     2,      .001,   0.9
DCONSTR,20,     3,      .001,   0.9
DCONSTR,20,     4,      .001,   0.9
DCONSTR,20,     5,      .001,   0.9
DCONSTR,20,     6,      .001,   0.9
DCONSTR,20,     7,      .001,   0.9
DCONSTR,20,     8,      .001,   0.9
$DOPTPRM1
7
20
0.3
DOPTPRM,IPRINT, 7,      DESMAX, 20,     DELP,   0.3,    CONV2,  1.0E-3
ENDDATA
$.......2.......3.......4.......5.......6.......7.......8.......9.......0
```

Listing 7-10.

7.10 Acoustic Optimization

Acoustic Optimization introduces acoustic pressures as a design response. These are computed from a solution of the coupled fluid-structure interaction problem. An optimal design can thus be found based not only on a consideration of acoustic pressures, but structural responses as well.

This example considers a closed box with fluid elements on the interior. An acoustic source is located at one end of the box, with a transducer located at the opposite end. The design goal is to modify the thicknesses of the box walls such that the peak acoustic pressure at the transducer is minimized. This is to be done without increasing the weight of the box.

The box geometry and property groups of thicknesses to be modified are shown in Figure 7-24. Six design variables are to be related to six of these property groups (the third property group in Figure 7-24 remains fixed.) This model has 1000 structural elements and 2000 fluid elements.

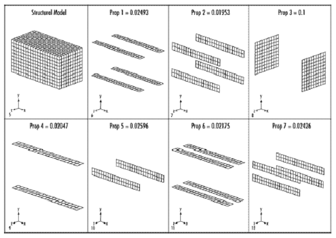


Figure 7-24. Acoustic Box Showing Portions Designed by Each Design Variable (Property 3 is Fixed)

An interesting feature of this, as well as many other dynamic response problems, is that a number of response peaks may exist. In addition, during the course of design optimization, these peaks may not only increase or decrease, but shift as well. This example presents a useful way of addressing this problem.

Figure 7-25 shows an arbitrary pressure distribution as a function of frequency, $P(f)$. Three frequencies of interest are shown as f_1, f_2, and f_3. What we would like to do is to minimize the maximum pressure response over all of these points.

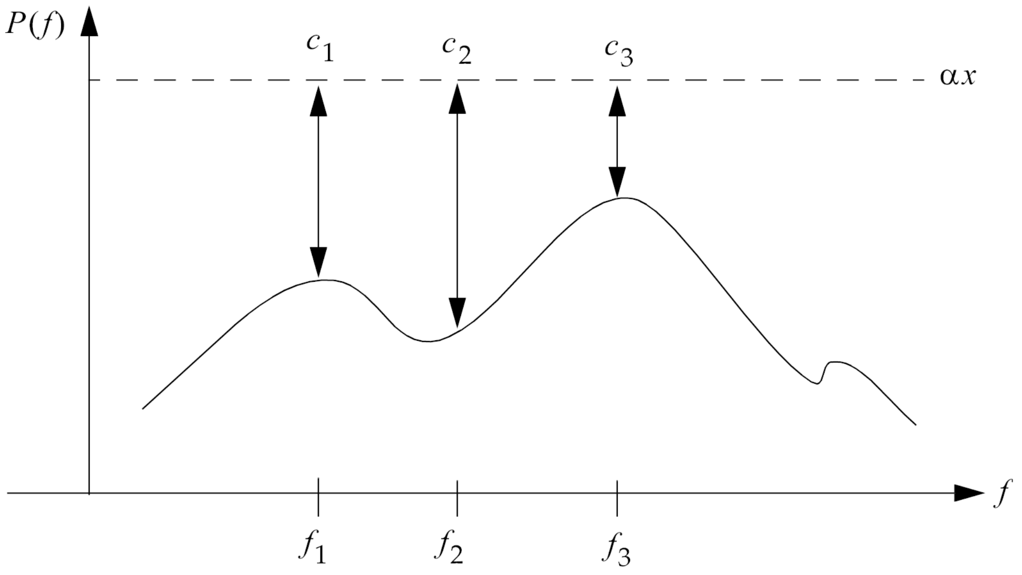


Figure 7-25. Minimization of Response Peaks

We can choose a design variable to represent a peak threshold, here shown in the figure as α*x* (α is just a constant of proportionality to facilitate scaling the threshold). The difference between α*x* and the pressure distribution must be a positive quantity at all frequencies of interest. Thus, constraints can be written that require c_1, c_2, and c_3 to be positive distances. The design objective can then be to minimize α*x*, or

minimize

$$\alpha x$$

Equation 7-22.

subject to:

$$\begin{aligned} P(f1) - \alpha x &\le 0 \\ P(f2) - \alpha x &\le 0 \\ P(f3) - \alpha x &\le 0 \end{aligned}$$

Equation 7-23.

As the optimizer decreases the threshold, the constraints ensure that the peaks are reduced as well. Any number of these constraints can be written to cover all frequency ranges of interest (or, in the case of transient analysis, time steps).

Due to the size of this problem, Listing 7-12 is an excerpt of the input file consisting mainly of the design model entries, as well as some the analysis entries. Analysis modeling for acoustics is covered in the *NX Nastran User's Guide* and in the *NX Nastran Advanced Dynamic Analysis User's Guide*.

Turning to the listing, we see that the weight budget is established as a global constraint. The weight response is defined on DRESP1 number 2; bounds are placed on this response with DCONSTR number 5, which is in turn referenced in Case Control by the DESGLB command. The bounds on weight allow less than one third of one percent variation from the initial weight. Allowing some variation is usually preferred over an equality constraint since it not only yields a better conditioned problem, but is often consistent with our design goals as well.

The objective function is defined as a constant times design variable number 8 (see Eq. 7-22, the objective function, and Eq. 7-24 below) using the DRESP2 number 100 entry. This refers to DEQATN,100 which defines

$$F = 10000. \cdot x_8$$

Equation 7-24.

This is defined as the objective function by the DESOBJ,100 Case Control command.

The constraints on response peaks are defined by first identifying the pressure responses themselves with DRESP1 entry number 1. This designates the "1" component of displacement at grid 11280 as the response. Since this grid is part of the fluid model, the "displacement" is actually a pressure with units of N/m^2. Since the ATTB field is blank on this DRESP1 entry, the response is computed for all output frequencies.

These pressure responses are used as input to DEQATN,10 via DRESP2 number 11. These entries, in combination with DCONSTR number 10, identify constraints on pressure response of the form:

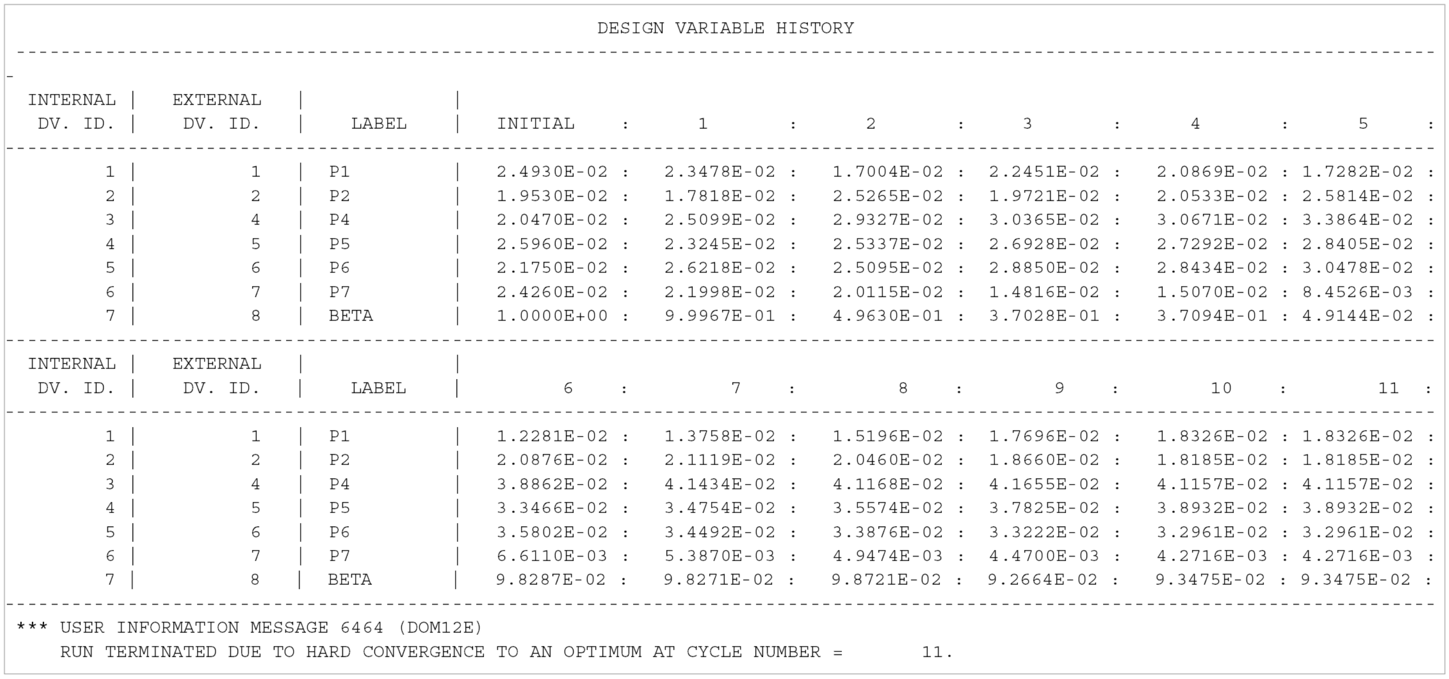
$$k_1 \cdot x_8 - P(f) + k_2 \geq k_2$$

Equation 7-25.

Note that this is similar to Eq. 7-23, with the constant k_2 added to avoid a bound of zero on the constraint. It should be clear that the constants have been chosen to scale the objective and responses to values that would minimize numerical difficulties and that there are really no "rules" for defining these factors.

Figure 7-26 and Figure 7-27 present the optimization results. The design cycle history of Figure 7-26 shows the objective has decreased from 10000.0 to 93.475 in eleven iterations. Figure 7-27 shows the magnitude of the frequency dependent pressure at the initial and final designs, and it is seen that there has been a dramatic reduction in the peak pressures from 140.5 dB to 115.0 dB (from 210.7 N/m^2 to 11.4 N/m^2).

```
                                 ***************************************************************
                                 S U M M A R Y   O F   D E S I G N   C Y C L E   H I S T O R Y
                                 ***************************************************************

                                                     (HARD CONVERGENCE ACHIEVED)

                                                     (SOFT CONVERGENCE ACHIEVED)
                                     NUMBER OF FINITE ELEMENT ANALYSES COMPLETED          12
                                     NUMBER OF OPTIMIZATIONS W.R.T. APPROXIMATE MODELS    11

                                            OBJECTIVE AND MAXIMUM CONSTRAINT HISTORY
---------------------------------------------------------------------------------------------------------------
                 OBJECTIVE FROM           OBJECTIVE FROM         FRACTIONAL ERROR         MAXIMUM VALUE
  CYCLE            APPROXIMATE                EXACT                     OF                      OF
  NUMBER          OPTIMIZATION              ANALYSIS              APPROXIMATION              CONSTRAINT
---------------------------------------------------------------------------------------------------------------

  INITIAL                                  1.000000E+04                                       1.106874E-01

        1           9.996712E+03           9.996712E+03            0.000000E+00              -2.604586E-03

        2           4.962984E+03           4.962984E+03            0.000000E+00               2.099861E-03

        3           3.702784E+03           3.702784E+03            0.000000E+00               1.015833E-02

        4           3.709424E+03           3.709424E+03            0.000000E+00               7.083985E-03

        5           4.914370E+02           4.914370E+02            0.000000E+00               2.863409E-02

        6           9.828741E+02           9.828741E+02            0.000000E+00               1.188324E-01

        7           9.827139E+02           9.827139E+02            0.000000E+00               2.536414E-02

        8           9.872102E+02           9.872102E+02            0.000000E+00               1.190643E-02

        9           9.266409E+02           9.266409E+02            0.000000E+00               9.985107E-03

       10           9.347542E+02           9.347542E+02            0.000000E+00               2.054260E-03

       11           9.347542E+02           9.347542E+02            0.000000E+00               2.054260E-03
---------------------------------------------------------------------------------------------------------------
```



```
                                                   DESIGN VARIABLE HISTORY
 --------------------------------------------------------------------------------------------------------------------------------
-
 INTERNAL |   EXTERNAL    |               |
 DV. ID.  |   DV. ID.     |     LABEL     |   INITIAL    :       1       :       2       :      3       :       4       :      5      :
---------------------------------------------------------------------------------------------------------------------------------
        1 |             1 |  P1           |   2.4930E-02 :   2.3478E-02 :   1.7004E-02 :  2.2451E-02 :   2.0869E-02 : 1.7282E-02 :
        2 |             2 |  P2           |   1.9530E-02 :   1.7818E-02 :   2.5265E-02 :  1.9721E-02 :   2.0533E-02 : 2.5814E-02 :
        3 |             4 |  P4           |   2.0470E-02 :   2.5099E-02 :   2.9327E-02 :  3.0365E-02 :   3.0671E-02 : 3.3864E-02 :
        4 |             5 |  P5           |   2.5960E-02 :   2.3245E-02 :   2.5337E-02 :  2.6928E-02 :   2.7292E-02 : 2.8405E-02 :
        5 |             6 |  P6           |   2.1750E-02 :   2.6218E-02 :   2.5095E-02 :  2.8850E-02 :   2.8434E-02 : 3.0478E-02 :
        6 |             7 |  P7           |   2.4260E-02 :   2.1998E-02 :   2.0115E-02 :  1.4816E-02 :   1.5070E-02 : 8.4526E-03 :
        7 |             8 |  BETA         |   1.0000E+00 :   9.9967E-01 :   4.9630E-01 :  3.7028E-01 :   3.7094E-01 : 4.9144E-02 :
---------------------------------------------------------------------------------------------------------------------------------
 INTERNAL |   EXTERNAL    |               |
 DV. ID.  |   DV. ID.     |     LABEL     |       6      :       7      :       8      :      9      :      10      :      11     :
---------------------------------------------------------------------------------------------------------------------------------
        1 |             1 |  P1           |   1.2281E-02 :   1.3758E-02 :   1.5196E-02 :  1.7696E-02 :   1.8326E-02 : 1.8326E-02 :
        2 |             2 |  P2           |   2.0876E-02 :   2.1119E-02 :   2.0460E-02 :  1.8660E-02 :   1.8185E-02 : 1.8185E-02 :
        3 |             4 |  P4           |   3.8862E-02 :   4.1434E-02 :   4.1168E-02 :  4.1655E-02 :   4.1157E-02 : 4.1157E-02 :
        4 |             5 |  P5           |   3.3466E-02 :   3.4754E-02 :   3.5574E-02 :  3.7825E-02 :   3.8932E-02 : 3.8932E-02 :
        5 |             6 |  P6           |   3.5802E-02 :   3.4492E-02 :   3.3876E-02 :  3.3222E-02 :   3.2961E-02 : 3.2961E-02 :
        6 |             7 |  P7           |   6.6110E-03 :   5.3870E-03 :   4.9474E-03 :  4.4700E-03 :   4.2716E-03 : 4.2716E-03 :
        7 |             8 |  BETA         |   9.8287E-02 :   9.8271E-02 :   9.8721E-02 :  9.2664E-02 :   9.3475E-02 : 9.3475E-02 :
---------------------------------------------------------------------------------------------------------------------------------
*** USER INFORMATION MESSAGE 6464 (DOM12E)
    RUN TERMINATED DUE TO HARD CONVERGENCE TO AN OPTIMUM AT CYCLE NUMBER =       11.
```

Figure 7-26. Design Cycle History for Acoustic Optimization Example

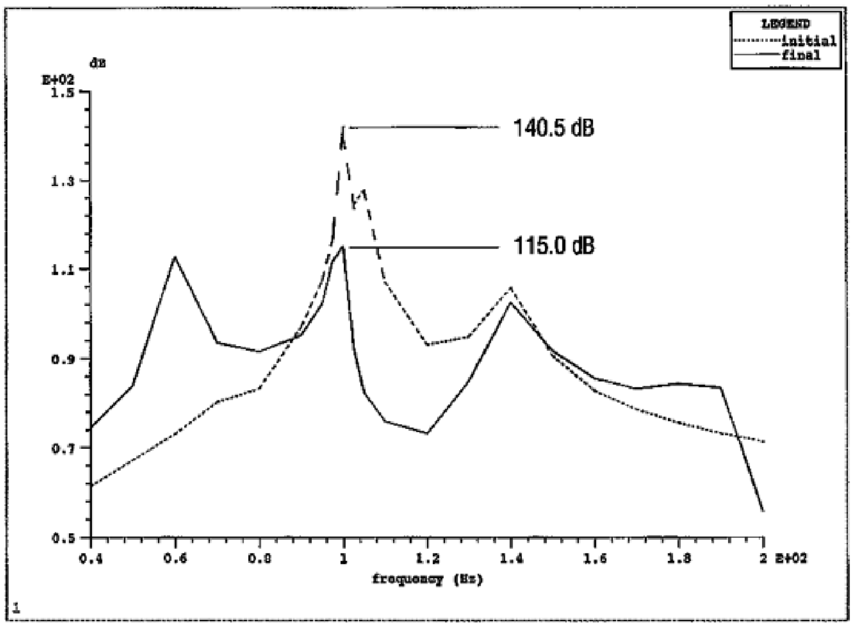


Figure 7-27. Acoustic Optimization Sound Pressure Levels: Initial and Final Distributions

Table 7-1 lists the first eight initial and final structural eigenvalues as well as the first nine invariant fluid eigenvalues. Note that the fluid has three repeated roots at 100.4 Hz and it is the coupling between these fluid resonances and the nearby structural resonances that creates the peak response. It is also notable that the final design has two structural frequencies near 60 Hz that are showing significant response.

Table 7-1. Structural and Fluid Eigenfrequencies in Hz

Mode No.	Structural Eigenfrequencie		Fluid Eigenfrequencies
	Initial	Final	
1	75.99	59.49	50.01
2	95.29	60.78	100.41
3	104.16	94.29	100.41
4	130.78	98.56	100.41
5	133.35	139.46	112.19
6	143.74	140.27	112.19
7	153.16	165.44	142.00
8	173.69	167.38	142.00
9	-	-	142.00

```
ID     UGS
TIME 100
SOL    200  $ modal frequency response
CEND
SUBTITLE = acoustic and structural elements
LABEL =   boxae1.dat
$
$ analysis case control:
set 20 = 11280
echo = sort(param,eigc,eigrl,freq,desvar,dconstr,dresp1,dresp2,deqatn,
            dvprel1)
spc = 1
DISP(phase) = 20
method(structure) = 20
method(fluid) = 30
cmethod = 10
frequency = 200
dload = 100
partn = 20
$ optimization case control:
ANALYSIS =  MFREQ
DESGLB   = 5
DESSUB   = 10
DESOBJ   = 100
$
BEGIN BULK
$
EIGRL   20              200.                                        max
eigrl   ,30,       15.,    155.,    9,      0,     ,     105.,      max
eigc,10,clan,max,,,,
,15.,300.,,,,,4
$
$ sound pressure level
param,rms,yes
$ reference pressure for dB and dBA
param,prefdb,2.-5
$
PARAM   AUTOSPC  no
$
$ fluid/structure interface
acmodl,diff ,  ,  ,  ,0.01
$
$ structural damping:
param,g,0.02
$
$-------2-------3-------4-------5-------6-------7-------8-------9-------10------
rload1,100,101,,,102
darea,101,1288,3,100.
tabled1,102
,0.,1.,1000.,1.,endt
$freq2, 200, 1.0, 200.0, 50
$-------2-------3-------4-------5-------6-------7-------8-------9-------10------
freq    200     40.     50.     60.     70.     80.     90.     95.
        100.    105.    110.    120.    130.    140.    150.    160.
        170.    180.    190.    200.     97.5   102.5
$-------2-------3-------4-------5-------6-------7-------8-------9-------10------
$
$
$ Define the Design Variables:
$
$ESVAR, ID,     LABEL,  XINIT,  XLB,    XUB,    DELXV
DESVAR, 1,      P1,     0.02493,0.0001, 1.
```

```
DESVAR, 2,      P2,     0.01953,0.0001, 1.
DESVAR, 4,      P4,     0.02047,0.0001, 1.
DESVAR, 5,      P5,     0.02596,0.0001, 1.
DESVAR, 6,      P6,     0.02175,0.0001, 1.
DESVAR, 7,      P7,     0.02426,0.0001, 1.
DESVAR 8 BETA 1.0 0.001
$
$ Relate the Design Variables to changes in plate thicknesses:
$
$VPREL1,ID,     TYPE,   PID,    FID,    PMIN,   PMAX,   C0,     ,       +
$+,     DVID1,  COEF1,  DVID2,  COEF2,  ...
$
DVPREL1,1,      PSHELL, 1,      4,      0.0001, ,       ,       ,       +
+,      1,      1.
DVPREL1,2,      PSHELL, 2,      4,      0.0001, ,       ,       ,       +
+,      2,      1.
DVPREL1,4,      PSHELL, 4,      4,      0.0001, ,       ,       ,       +
+,      4,      1.
DVPREL1,5,      PSHELL, 5,      4,      0.0001, ,       ,       ,       +
+,      5,      1.
DVPREL1,6,      PSHELL, 6,      4,      0.0001, ,       ,       ,       +
+,      6,      1.
DVPREL1,7,      PSHELL, 7,      4,      0.0001, ,       ,       ,       +
+,      7,      1.
$
$ Define the synthetic Objective as a function of X8:
$$RESP2, ID,     LABEL,  EQID,   REGION, ,       ,       ,       ,       +
$+,     DESVAR, DVID1,  DVID2,  ...,    ,       ,       ,       ,       +
DRESP2 100 BETA 100
 DESVAR 8
DEQATN  100  OBJ(BETA) = 10000.0 * BETA
$
$ Define the constraint on weight (weight budget for optimization):
$
$RESP1, ID,     LABEL,  RTYPE,  PTYPE,  REGION, ATTA,   ATTB,   ATT1,   +
$+,     ATT2,   ...
$
DRESP1  2       WEIGHT  WEIGHT
$CONSTR,DCID,   RID,    LALLOW, UALLOW
DCONSTR 5 2 2890. 2910.
$
$ Define the constraints on acoustic sound pressure levels
$ (one for each forcing frequency):
$
DRESP1  1       DRUCK   FRDISP                  1               11280
DRESP2 11 PRESBET 10
 DESVAR 8
        DRESP1  1
DEQATN 10 F(BETA,PRES) = 100.0 * BETA - PRES + 1000.
DCONSTR 10 11 1000.
$
$ Override miscellaneous Optimization Parameters:
$
DOPTPRM DESMAX 20 P1 1 P2 15 CONV1 1E-6
$
$
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$       structural model
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$
$
$ THIS SECTION CONTAINS THE PROPERTY AND MATERIAL BULK DATA ENTRIES
$
PSHELL  3       1       .100    1
$
PSHELL  1       1       .02493  1
PSHELL  2       1       .01953  1
pshell  4       1       .02047  1
pshell  5       1       .02596  1
pshell  6       1       .02175  1
pshell  7       1       .02426  1
$
$
$ THIS SECTION CONTAINS BULK DATA FOR SUPERELEMENT 0
$
$
GRID    1               0.0     0.0     0.0
GRID    2               2.      0.0     0.0
GRID    3               2.      0.0     1.
  .     .               .       .       .
  .     .               .       .       .
  .     .               .       .       .
GRID    1293            1.5     .5      1.
GRID    1294            1.6     .5      1.
GRID    1295            1.7     .5      1.
$
CQUAD4  1       1       1       9       29      28
CQUAD4  2       1       9       10      30      29
   .    .       .       .       .       .       .
   .    .       .       .       .       .       .
   .    .       .       .       .       .       .
CQUAD4  999     3       1139    1140    275     296
```

```
CQUAD4  1000    3       1140    897     5       275
$
$ THIS SECTION CONTAINS THE LOADS,CONSTRAINTS, AND CONTROL BULK DATA ENTRIES
$
$
SPC     1       1       123     0.0
SPC     1       2       123     0.0
   .    .       .       .       .
   .    .       .       .       .
   .    .       .       .       .
SPC            1     986       4
SPC            1     987       4
SPC            1     975       4
$
$
MAT1    1       2.+11           .3      7600.
$
$-------2-------3-------4-------5-------6-------7-------8-------9-------10------
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$       acoustic model
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$
$ THIS SECTION CONTAINS BULK DATA FOR SUPERELEMENT 0
$
$
$-------2-------3-------4-------5-------6-------7-------8-------9-------10------
GRID    10001           0.0     0.0     0.0           -1
GRID    10002           2.      0.0     0.0           -1
GRID    10003           2.      0.0     1.            -1
   .      .             .       .       .              .
   .      .             .       .       .              .
   .      .             .       .       .              .
GRID    12540           2.      1.      .2            -1
GRID    12541           2.      1.      .1            -1
$
CHEXA      10001100     10004   10126   10127   10009   10018   10137   +
+       10138   10019
CHEXA      10002100     10009   10127   10128   10010   10019   10138   +
+       10139   10020
   .      .     .       .       .       .       .       .       .
   .      .     .       .       .       .       .       .       .
   .      .     .       .       .       .       .       .       .
CHEXA      11999100     12411   12530   12531   12412   12422   12540   +
+       12541   12423
CHEXA      12000100     12412   12531   12532   12413   12423   12541   +
+       10006   12424
$
psolid,100,100,,,,1,pfluid
$
mat10,100,,1.293,200.
$
ENDDATA
```

Listing 7-11.

7.11 Restarts in Design Optimization

In design optimization, two types of restarts are possible. The first should probably be called a "pseudo-restart" since it is simply a cold start using the results of the previous design. The second type is more of a true restart, in the NX Nastran sense of the word, since it allows results saved on the database to be used by a subsequent job. This feature is somewhat limited, however, in design optimization.

Probably the best way to illustrate these restart features is by example. This section describes both types of restarts and discusses some of the limitations of each type.

Restarting from a Previous Design Cycle

If the design and analysis models differ in Solution 200, the design model data is used to override the analysis model. This feature can be used to our advantage to perform a simple type of restart. For example, if the initial design variable values are modified on the DESVAR entries, the code uses this as the "initial" design for the subsequent optimization. This set might have been determined based on a prior optimization cycle or may reflect other changes that the engineer might want to make.

For every design cycle completed, a "punch" file is created (.PCH) that contains the DESVAR Bulk Data entries for this latest design. The contents of the punch file can be included in the original input deck to "restart" from this point in the design space.

Using the venerable Three-Bar Truss as an example (TPL deck D200X1.DAT), suppose that we want to perform a single design cycle with some of the optimization parameters overridden, examine these results, readjust the parameters, and continue with optimization. Listing 7-12 shows the input data file for the first design cycle.

Note

The shape optimization capabilities in the software also include output of the updated GRID entries. The parameter DESPCH controls the frequency of both updated GRID and DESVAR entry output to the punch file. See Parameters for Design Sensitivity and Optimization .

Since displacement and stress responses vary inversely to the bar cross-sectional dimensions, APRCOD = 2 or the mixed method of approximations is selected on the DOPTPRM entry. (APRCOD = 2 is the default anyway, but this makes it explicit in the input data). Since this is likely to yield a good approximation for truss-type structures, we have increased the move limits to 100% for this first design cycle (DELP = 1.0). With these relatively wide move limits, we may want to make sure that all of the constraints are included, just to be sure. DSCREEN entries have been included to reduce the truncation threshold TRS to a large negative number to ensure that none of the displacement or stress constraints are screened out prior to the approximate optimization.

The design cycle history is shown in Figure 7-28. Compare these results to the original Three-Bar Truss example in Three-Bar Truss. Note that the 100% move limits allow a greater move in x_2 for the first cycle, which results in a somewhat lower value of the objective. (In fact, notice that x_2 actually changed by slightly more than 50% in the original problem. As mentioned in Optimization with Respect to Approximate Models , this may occur since move limits are imposed as equivalent constraints on the corresponding properties. Critical constraints may not be satisfied exactly, as is seen to be the case here. This is usually inconsequential, since this design just serves as a starting point for the next optimization cycle.)

```
                                     ***************************************************************
                                     S U M M A R Y   O F   D E S I G N   C Y C L E   H I S T O R Y
                                     ***************************************************************

                                         NUMBER OF FINITE ELEMENT ANALYSES COMPLETED           2
                                         NUMBER OF OPTIMIZATIONS W.R.T. APPROXIMATE MODELS     1

                                                OBJECTIVE AND MAXIMUM CONSTRAINT HISTORY
                  ---------------------------------------------------------------------------------------------------------------
                                     OBJECTIVE FROM           OBJECTIVE FROM          FRACTIONAL ERROR          MAXIMUM VALUE
                    CYCLE             APPROXIMATE                 EXACT                     OF                      OF
                    NUMBER            OPTIMIZATION              ANALYSIS              APPROXIMATION             CONSTRAINT
                  ---------------------------------------------------------------------------------------------------------------

                    INITIAL                                   4.828427E+00                                    -3.234952E-01

                          1            2.888302E+00           2.888885E+00           -2.020325E-04            -2.564473E-02
                  ---------------------------------------------------------------------------------------------------------------

                                                           DESIGN VARIABLE HISTORY
        ------------------------------------------------------------------------------------------------------------------------------
         INTERNAL |   EXTERNAL    |              |
          DV. ID. |    DV. ID.    |     LABEL    |   INITIAL    :       1        :       2        :       3        :       4        :       5     :
        ------------------------------------------------------------------------------------------------------------------------------
                1 |          1    |  A1          |   1.0000E+00 :   7.6865E-01 :
                2 |          2    |  A2          |   2.0000E+00 :   7.1480E-01 :
                3 |          3    |  A3          |   1.0000E+00 :   7.6865E-01 : ------------------------------------------------------------
       ------------------------------------------------------------------
        *** USER INFORMATION MESSAGE 6464 (DOM12E)
            RUN TERMINATED DUE TO MAXIMUM NUMBER OF DESIGN CYCLES =        1.
```

Figure 7-28. Single Design Cycle

Turning now to the pseudo-restart, the punch file from the previous design cycle is included in the restart deck listed in Listing 7-13. Only the design model portion of the deck has been included in Listing 7-13, since the analysis model portion is identical to Listing 7-12. The optimization now begins from the point in the design space defined by the new DESVAR entries. The DOPTPRM and DSCREEN entries are updated in this deck as well to allow more design iterations (DESMAX), reduce the move limits to 50% (DELP), and set the truncation threshold back to -0.5 (TRS). (Leaving out the DSCREEN entries altogether accomplishes the same thing since this is the TRS default.)

The optimization results from this input are listed in Figure 7-29, which are seen to be slightly better than the original D200X1. If this job were much larger and more complex, the ability to pause, examine the results, and readjust some of the optimization parameters could be indispensable.

```
                                   ***************************************************************
                                   S U M M A R Y   O F   D E S I G N   C Y C L E   H I S T O R Y
                                   ***************************************************************

                                                 (HARD CONVERGENCE ACHIEVED)

                                       NUMBER OF FINITE ELEMENT ANALYSES COMPLETED           5
                                       NUMBER OF OPTIMIZATIONS W.R.T. APPROXIMATE MODELS     4

                                              OBJECTIVE AND MAXIMUM CONSTRAINT HISTORY
           ---------------------------------------------------------------------------------------------------------------
                             OBJECTIVE FROM         OBJECTIVE FROM        FRACTIONAL ERROR        MAXIMUM VALUE
             CYCLE             APPROXIMATE              EXACT                   OF                    OF
             NUMBER           OPTIMIZATION             ANALYSIS            APPROXIMATION          CONSTRAINT
           ---------------------------------------------------------------------------------------------------------------

             INITIAL                                2.888885E+00                                 -2.564473E-02

                   1            2.752784E+00        2.752907E+00          -4.460214E-05          -7.020019E-03

                   2            2.719121E+00        2.719218E+00          -3.551004E-05          -4.937890E-03

                   3            2.697654E+00        2.697639E+00           5.479589E-06           1.350586E-03

                   4            2.694388E+00        2.694371E+00           6.105647E-06           2.292285E-03
           ---------------------------------------------------------------------------------------------------------------

                                                       DESIGN VARIABLE HISTORY
 ------------------------------------------------------------------------------------------------------------------------------
  INTERNAL |   EXTERNAL   |             |
   DV. ID. |    DV. ID.   |    LABEL    |   INITIAL    :      1       :      2       :      3       :      4       :   5       :
 ------------------------------------------------------------------------------------------------------------------------------
         1 |          1   |  A1         |   7.6865E-01 :   7.9464E-01 :   8.2615E-01 :   8.3462E-01 :   8.3819E-01 :
         2 |          2   |  A2         |   7.1480E-01 :   5.0532E-01 :   3.8250E-01 :   3.3696E-01 :   3.2361E-01 :
         3 |          3   |  A3         |   7.6865E-01 :   7.9464E-01 :   8.2615E-01 :   8.3462E-01 :   8.3819E-01 : ------------
 ------------------------------------------------------------------------------------------------------------------
 0*** USER INFORMATION MESSAGE 6464 (DOM12E)
      RUN TERMINATED DUE TO HARD CONVERGENCE TO AN OPTIMUM AT CYCLE NUMBER =        4.
```

Figure 7-29. Optimization from Updated Design

Looking back on the summary of design cycle history for the first run, note that two finite element analyses were performed. The first provided the baseline analysis, while the second updated the set of responses according to the proposed design changes. The first task of the subsequent "restart" repeats this analysis, though, to provide the new, baseline responses. If the analysis cost is high, this repeated analysis might be troublesome. The costs associated with this type of restart must be weighed against its convenience.

Restarting from an OPTEXIT Point

The second class of restarts is based on NX Nastran's automatic restart capability. In design optimization, this works in conjunction with the predefined Solution 200 exit points, OPTEXIT 1 through 6. Recall that setting this parameter in Bulk Data will terminate the program at one of these six exit points.

Automatic restart for design optimization is somewhat limited since only the analysis restart logic is in place. That is, only data associated with the analysis results are used on restart, and all design optimization-related processes are repeated. For example, restarts from an OPTEXIT point of 4 or greater will result in a repeat of the sensitivity analysis phases. This occurs regardless of whether the design model has changed or not. (Warning: the design model must not be changed, as unpredictable results may occur.)

To illustrate this procedure, suppose we wanted to perform a baseline analysis for the structure, compute the sensitivities of the design responses with respect to changes in the design variables, and then pause to examine these results before restarting. The results of the finite element analysis are saved to the database by including SCR=NO in the job submittal command.

The only difference between the original three-bar truss and the input here is the Bulk Data parameter OPTEXIT. In this case, OPTEXIT is set to 4, which instructs the code to perform sensitivity analysis, print out the coefficients, and stop.

After the finite element and sensitivity analysis results are examined and verified, the decision is made to continue with optimization in the restart run. This deck is shown in Listing 7-14.

The RESTART command is required, and indicates that the most recent version on the database is to be used (VERSION=LAST), and only the latest data is to be retained (NOKEEP). If we always intend to use the first version on the database for restarts, we can instead use,

```
RESTART, VERSION=1, KEEP
```

This command appends the subsequent results to the database while saving the data from prior runs. This might be useful if the restart yields results that we do not want to save and we want to go back and restart from the initial analysis. By specifying VERSION=1, KEEP, we always have the first version to fall back on.

The command,

```
ASSIGN MASTER=d200x1c.MASTER
```

assigns the correct DBSET, so we do not have to repeatedly assign this with `DBS = ' in the job submittal command. (The lower-case filename ensures consistency with the filenaming on UNIX-based machines. On some machines, quote marks may also be required around the filename.)

` /,38 ' deletes the OPTEXIT parameter assignment from the Bulk Data, which is entry number 38 in the sorted data. It is usually a good idea to request a listing of the sorted Bulk Data, especially if subsequent restarts are anticipated. It is the default, anyway, but is explicit with the Case Control command, ECHO=SORT.

The optimization results from this restart are shown in Figure 7-30. Note that they agree with the results of the original Three-Bar Truss. The contents of the .F04 file (not included here) indicate the initial analysis was skipped.

```
                                              ***************************************************************
                                              S U M M A R Y   O F   D E S I G N   C Y C L E   H I S T O R Y
                                              ***************************************************************

                                                                   (HARD CONVERGENCE ACHIEVED)

                                                  NUMBER OF FINITE ELEMENT ANALYSES COMPLETED           6
                                                  NUMBER OF OPTIMIZATIONS W.R.T. APPROXIMATE MODELS     5

                                                         OBJECTIVE AND MAXIMUM CONSTRAINT HISTORY
           ---------------------------------------------------------------------------------------------------------------
                              OBJECTIVE FROM           OBJECTIVE FROM           FRACTIONAL ERROR           MAXIMUM VALUE
             CYCLE             APPROXIMATE                 EXACT                      OF                        OF
             NUMBER            OPTIMIZATION               ANALYSIS               APPROXIMATION              CONSTRAINT
           ---------------------------------------------------------------------------------------------------------------

             INITIAL                                     4.828427E+00                                        -3.234952E-01

                   1             3.007897E+00            3.008492E+00            -1.977251E-04               -3.737402E-03

                   2             2.821953E+00            2.821638E+00             1.118734E-04               -1.967246E-02

                   3             2.734469E+00            2.734299E+00             6.217039E-05               -7.241016E-03

                   4             2.708915E+00            2.708921E+00            -2.024285E-06               -2.369141E-04

                   5             2.702065E+00            2.702063E+00             7.941219E-07               -2.666992E-04
           ---------------------------------------------------------------------------------------------------------------

                                                                    DESIGN VARIABLE HISTORY ------------------------------------------
-----------------------------------------------------------------------------
  INTERNAL |   EXTERNAL   |           |
   DV. ID. |    DV. ID.   |    LABEL  |   INITIAL    :      1       :      2       :      3       :      4       :      5    : --
-----------------------------------------------------------------------------------------------------------------------------
         1 |          1   |  A1       |   1.0000E+00 :   7.1020E-01 :   7.8436E-01 :   8.1374E-01 :   8.1739E-01 :  8.3569E-01 :
         2 |          2   |  A2       |   2.0000E+00 :   9.9976E-01 :   6.0313E-01 :   4.3271E-01 :   3.9699E-01 :  3.3838E-01 :
         3 |          3   |  A3       |   1.0000E+00 :   7.1020E-01 :   7.8436E-01 :   8.1374E-01 :   8.1739E-01 :  8.3569E-01 :
------------------------------------------------------------------------------------------------------------------------------
0*** USER INFORMATION MESSAGE 6464 (DOM12E)
     RUN TERMINATED DUE TO HARD CONVERGENCE TO AN OPTIMUM AT CYCLE NUMBER =        5.
```

Figure 7-30. Restart Results from OPTEXIT = 4

```
ID UGS, D200X1
TIME  10      $
SOL 200       $  OPTIMIZATION
CEND
TITLE = SYMMETRIC THREE BAR TRUSS DESIGN OPTIMIZATION  -  D200X1
SUBTITLE = BASELINE - 2 CROSS SECTIONAL AREAS AS DESIGN VARIABLES
ECHO   = SORT
SPC    = 100
DISP   = ALL
STRESS = ALL
DESOBJ(MIN) = 20 $  (DESIGN OBJECTIVE = DRESP ID)
DESSUB      = 21 $  DEFINE CONSTRAINT SET FOR BOTH SUBCASES
ANALYSIS = STATICS
SUBCASE 1
   LABEL = LOAD CONDITION 1
   LOAD  = 300
SUBCASE 2
   LABEL = LOAD CONDITION 2
   LOAD  = 310
BEGIN BULK
$
$-----------------------------------------------------------------------
$ ANALYSIS MODEL
$-----------------------------------------------------------------------
$
$ GRID DATA
$       2       3       4       5       6       7       8       9        10
GRID,   1,      ,       -10.0 ,   0.0,  0.0
GRID,   2,      ,         0.0 ,   0.0,  0.0
GRID,   3,      ,        10.0 ,   0.0,  0.0
GRID,   4,      ,         0.0 , -10.0,  0.0
$ SUPPORT DATA
```

```
SPC,    100,    1,      123456, ,       2,      123456
SPC,    100,    3,      123456, ,       4,      3456
$ ELEMENT DATA
CROD,   1,      11,     1,      4
CROD,   2,      12,     2,      4
CROD,   3,      13,     3,      4
$ PROPERTY DATA
PROD,   11,     1,      1.0
PROD,   12,     1,      2.0
PROD,   13,     1,      1.0
MAT1,   1,      1.0E+7, ,       0.33,   0.1
$ EXTERNAL LOADS DATA
FORCE,  300,    4,      ,       20000.,  0.8,   -0.6
FORCE,  310,    4,      ,       20000., -0.8,   -0.6
$
$------------------------------------------------------------------------
$ DESIGN MODEL
$------------------------------------------------------------------------
$
$...DESIGN VARIABLE DEFINITION
$DESVAR,ID,     LABEL,  XINIT,  XLB,    XUB,    DELXV(OPTIONAL)
DESVAR, 1,      A1,     1.0,    0.1,    100.0
DESVAR, 2,      A2,     2.0,    0.1,    100.0
DESVAR, 3,      A3,     1.0,    0.1,    100.0
$
$...IMPOSE X3=X1 (LEADS TO A3=A1)
$DLINK, ID,     DDVID,  CO,     CMULT,  IDV1,   C1,     IDV2,   C2,     +
$+,     IDV3,   C3,     ...
DLINK,  1,      3,      0.0,    1.0,    1,      1.00
$
$...DEFINITION OF DESIGN VARIABLE TO ANALYSIS MODEL PARAMETER RELATIONS
$DVPREL1,ID,    TYPE,   PID,    FID,    PMIN,   PMAX,   C0,     ,       +
$+,     DVID1,  COEF1,  DVID2,  COEF2,  ...
DVPREL1,10,     PROD,   11,     4,      ,       ,       ,       ,       +DP1
+DP1,   1,      1.0
DVPREL1,20,     PROD,   12,     4,      ,       ,       ,       ,         +DP2
+DP2,   2,      1.0
DVPREL1,30,     PROD,   13,     4,      ,       ,       ,       ,         +DP3
+DP3,   3,      1.0
$
$...STRUCTURAL RESPONSE IDENTIFICATION
$DRESP1,ID,     LABEL,  RTYPE,  PTYPE,  REGION, ATTA,   ATTB,   ATT1,   +
$+,     ATT2,   ...
DRESP1, 20,     W ,     WEIGHT
DRESP1, 21,     U4,     DISP  , ,       ,       1,      ,       4
DRESP1, 22,     V4,     DISP  , ,       ,       2,      ,       4
DRESP1, 23,     S1,     STRESS, PROD,   ,       2,      ,       11
DRESP1, 24,     S2,     STRESS, PROD,   ,       2,      ,       12
DRESP1, 25,     S3,     STRESS, PROD,   ,       2,      ,       13
$
$...CONSTRAINTS
$DCONSTR,DCID,  RID,    LALLOW, UALLOW
DCONSTR,21,     21,     -0.20  ,0.20
DCONSTR,21,     22,     -0.20  ,0.20
DCONSTR,21,     23,     -15000.,20000.
DCONSTR,21,     24,     -15000.,20000.
DCONSTR,21,     25,     -15000.,20000.
$
$...OPTIMIZATION CONTROL:
$
DSCREEN,DISP,   -100.,  10
DSCREEN,STRESS, -100.,  10
DOPTPRM,APRCOD, 2,      IPRINT, 5,      DESMAX, 1,      DELP,   1.0,    +
+,      P1,     1,      P2,     15
$
$.......2.......3.......4.......5.......6.......7.......8.......9.......0
ENDDATA
$
```

Listing 7-12.

```
$------------------------------------------------------------------------
$ DESIGN MODEL
$------------------------------------------------------------------------
$
$...DESIGN VARIABLE DEFINITION
$DESVAR,ID,     LABEL,  XINIT,  XLB,    XUB,    DELXV(OPTIONAL)
DESVAR *               1A1                 7.68653870E-01  1.00000001E-01+D    1V
*D    1V  1.00000000E+02  1.00000000E+00
DESVAR *               2A2                 7.14803874E-01  1.00000001E-01+D    2V
*D    2V  1.00000000E+02  1.00000000E+00
DESVAR *               3A3                 7.68653870E-01  1.00000001E-01+D    3V
*D    3V  1.00000000E+02  1.00000000E+00
$
$...IMPOSE X3=X1 (LEADS TO A3=A1)
$DLINK, ID,     DDVID,  CO,     CMULT,  IDV1,   C1,     IDV2,   C2,     +
$+,     IDV3,   C3,     ...
DLINK,  1,      3,      0.0,    1.0,    1,      1.00
$
$...DEFINITION OF DESIGN VARIABLE TO ANALYSIS MODEL PARAMETER RELATIONS
$DVPREL1,ID,    TYPE,   PID,    FID,    PMIN,   PMAX,   C0,     ,       +
$+,     DVID1,  COEF1,  DVID2,  COEF2,  ...
DVPREL1,10,     PROD,   11,     4,      ,       ,       ,       ,       +DP1
+DP1,   1,      1.0
DVPREL1,20,     PROD,   12,     4,      ,       ,       ,       ,         +DP2
+DP2,   2,      1.0
DVPREL1,30,     PROD,   13,     4,      ,       ,       ,       ,         +DP3
+DP3,   3,      1.0
$
$...STRUCTURAL RESPONSE IDENTIFICATION
$DRESP1,ID,     LABEL,  RTYPE,  PTYPE,  REGION, ATTA,   ATTB,   ATT1,   +
$+,     ATT2,   ...
DRESP1, 20,     W ,     WEIGHT
DRESP1, 21,     U4,     DISP  , ,       ,       1,      ,       4
DRESP1, 22,     V4,     DISP  , ,       ,       2,      ,       4
DRESP1, 23,     S1,     STRESS, PROD,   ,       2,      ,       11
DRESP1, 24,     S2,     STRESS, PROD,   ,       2,      ,       12
DRESP1, 25,     S3,     STRESS, PROD,   ,       2,      ,       13
$
$...CONSTRAINTS
$DCONSTR,DCID,  RID,    LALLOW, UALLOW
DCONSTR,21,     21,     -0.20  ,0.20
DCONSTR,21,     22,     -0.20  ,0.20
DCONSTR,21,     23,     -15000.,20000.
DCONSTR,21,     24,     -15000.,20000.
DCONSTR,21,     25,     -15000.,20000.
$
$...OPTIMIZATION CONTROL:
$
DSCREEN,DISP,   -.5
DSCREEN,STRESS, -.5
DOPTPRM,IPRINT, 5,      DESMAX, 10,     DELP,   0.5,    P1,     1,      +
+,      P2,     15
$
$.......2.......3.......4.......5.......6.......7.......8.......9.......0
ENDDATA
```

Listing 7-13.

```
RESTART, VERSION=LAST, NOKEEP
ASSIGN MASTER=d200x1c.MASTER
ID EDS, D200X1
TIME  10      $
SOL 200       $  OPTIMIZATION
CEND
TITLE = SYMMETRIC THREE BAR TRUSS DESIGN OPTIMIZATION  -  D200X1
SUBTITLE = BASELINE - 2 CROSS SECTIONAL AREAS AS DESIGN VARIABLES
ECHO   = SORT
SPC    = 100
DISP   = ALL
STRESS = ALL
DESOBJ(MIN) = 20 $  (DESIGN OBJECTIVE = DRESP ID)
DESSUB      = 21 $  DEFINE CONSTRAINT SET FOR BOTH SUBCASES
ANALYSIS = STATICS
SUBCASE 1
   LABEL = LOAD CONDITION 1
   LOAD  = 300
SUBCASE 2
   LABEL = LOAD CONDITION 2
   LOAD  = 310
BEGIN BULK
/,34
ENDDATA
```

Listing 7-14.

Appendix A: Glossary of Terms

Active Constraint a constraint whose numerical value is near zero. In NX Nastran, the constant CT (default = –0.03) is used to measure whether a constraint is active or inactive. A constraint whose value is less than CT is deemed inactive, and if its value is greater than CT, it is considered active. This information is used at the optimizer level in connection with the numerical search process. See Violated Constraints.

Analysis Discipline is used in Solution 200 to refer to the available analysis types: Statics, Normal Modes, Buckling, Direct and Modal Frequency, Modal Transient, Static Aeroelastic Responses, and Flutter.

Analysis Model defines the geometry, element connectivity, material properties, and loads associated with an NX Nastran finite element analysis. The analysis model may be varied according to the design model, which uses responses computed from the analysis model to guide the design process.

Analysis Model Properties are, for purposes of design sensitivity and optimization in NX Nastran, as quantities specified on property Bulk Data entries. For example, bar area moments of inertia (on the PBAR entry) are analysis model properties, whereas offsets (on the CBAR entry) are not.

Approximate Model is an approximate representation of the design space that is used by the optimizer to determine an approximate optimum. Since the approximate model contains a reduced number of constraints and is explicit in the design variables, it lends itself to efficient solution by a numerical optimizer.

Approximate Optimization is used to refer to the optimization with respect to approximate design spaces. See Approximation Concepts in Design Optimization .

Approximation Concepts include design variable linking, constraint regionalization and deletion, and formal approximations. All of these approximations are available in NX Nastran. These tools allow numerical optimizers to be coupled effectively with structural analysis codes where the responses are usually implicit functions of the design variables. Solving the resultant approximate subproblem yields an approximate optimum.

Auxiliary Boundary Model is similar to an Auxiliary Model for generation of shape basis vectors, though defined just on the boundary of the finite element model. It may be, for example, an assembly of bar elements along the edge of a two-dimensional structure, or a group of plate elements over the surface of a three-dimensional model. It is used to generate the boundary portion of the shape basis vector from which the motion of interior points, and the remaining portion of the basis vector, is interpolated.

Auxiliary Model (or Auxiliary Structure) is a finite element model used to generate grid variations for purposes of shape sensitivity or optimization. It usually shares geometry and connectivity with the original (or primary) structure, but with different loads, boundary conditions, and perhaps material types. The resultant deformations are used to generate a basis vector for shape sensitivity and optimization.

Basic Optimization Problem Statement is a formal definition of the optimization task. This problem statement is invoked by the engineer when constructing the design model and by the optimizer when searching for an optimal design. See Eq. 1-1 through Eq. 1-5.

Basis Vector is a collection of constant terms used to relate the changes in a design variable to changes in structural properties. They may be used in connection with either property optimization or shape optimization. When used for shape optimization, they are commonly referred to as Shape Basis Vectors.

Central Difference Approximations are used to approximate derivatives, or sensitivities, of continuous functions. Central Difference Approximations, which are second order, yield greatly improved derivative approximations when compared to first order differencing schemes. (Forward and backward differences are first order.)

Constraint (Design Constraint) is defined as an inequality that must be satisfied to indicate a feasible design. In NX Nastran, the constraint may be a function of design variables, structural responses, and grid coordinates.

Constraint Deletion is the process that temporarily removes constraints for a given design cycle. Once a constraint is removed, its sensitivities do not need to be computed, and the optimizer does not need to consider the constraint during the approximate optimization. Later design cycles may include constraints that had been previously deleted.

Constraint Regionalization refers to the grouping of constraints by type in preparation for constraint deletion.

Constraint Screening is a type of approximation concept that seeks to reduce the number of constraints in the optimization problem. This reduction is based on the assumption that the entire constraint set is likely to contain redundant design information. Both regionalization and deletion are used to group and screen constraints for temporary deletion. Constraints that are temporarily deleted may be retained on subsequent design cycles.

Dependent Design Variable is a design variable defined on a DESVAR entry and is linearly-dependent on one or more independent design variables. This dependence is defined by DLINK Bulk Data entries.

Design Cycle refers to the process that invokes the optimizer once for each design cycle. Within a design cycle, a finite element analysis is performed, an approximate problem is created, the optimizer is invoked, and convergence is tested. A number of design cycles may need to be completed before overall convergence is achieved.

Design Iteration is the optimizer level process of determining a search direction from available gradient information, and performing a one-dimensional search in this direction. A number of these iterations may be performed before an approximate optimum is found.

Design Model defines the design variables, objective, and constraints. In the design model, design variables are used to express permissible analysis model variations, and design responses are used as the objective and constraint functions. The optimizer uses the information in the design model to propose improved designs.

Design Optimization refers to the automated redesign process that attempts to minimize a specific quantity, subject to limits or constraints on other responses. This process is often referred to as Structural Optimization in this guide to indicate the application of these methods to structural redesign tasks.

Design Sensitivity Analysis is the process that is used to determine rates of change of structural responses with respect to changes in design variables. The resulting partial derivatives or design sensitivity coefficients $\partial r_j/\partial x_i$ can be used directly to perform parametric design studies or as input to a numerical optimizer for design optimization.

Design Sensitivity Coefficient see Design Sensitivity Analysis.

Design Space is the *n*-dimensional region over which the objective and constraints are defined where *n* is the number of independent design variables. It is within this space that the optimizer searches for an optimal design, while simultaneously trying to satisfy all of the inequality constraints (an optimal, feasible solution).

Design Variable Linking allows a design variable to be expressed as a linear function of other, independent, design variables. This linking must be explicitly defined by the engineer. It is one of the components in the family of approximation concepts. Design variable linking can be used to enforce symmetry, ensure equivalent element properties, and so on. It can also improve the convergence characteristics of the problem by reducing the number of independent design variables that must be considered by the optimizer.

Designed Coordinates are grid coordinates that are allowed to vary in shape optimization. If a designed grid is to only move in the x-y plane, for example, then it will have only two designed coordinates x and y. The z-coordinate for that grid will not be designed. More specifically, the designed coordinates correspond to non-zero entries in the basis vector matrix for shape optimization.

Designed Grids are those grids whose locations vary in shape optimization. Grids on a moving boundary would thus be designed, as would be grids on the interior of the structure that are updated. Grids on a fixed boundary (one that does not change during shape optimization) would not be Designed Grids. See also Designed Coordinates.

Direct Input of Shapes refers to the method of DBLOCATEing displacement vectors that have been generated using an external Auxiliary Model. Once input, they may be used for shape sensitivity and optimization.

Feasible Design is a design that satisfies all of the constraints. A feasible design may not be optimal, though.

Finite Differences are a numerical method for approximating derivatives. The method derives from the Taylor series expansions of arbitrary functions. First-forward finite differences (see Eq. 1-9 and central differences (see Eq. 3-28, for example) are frequently used in NX Nastran's semianalytic sensitivity analysis to provide low-cost derivative approximations.

First-level Response is a response that is directly available from an NX Nastran analysis. These responses are identified for use in the design model with DRESP1 entries.

Formal Approximations are the Taylor series expansions of the implicit responses used in connection with design optimization. The resultant explicit approximations are used by the optimizer to find a corresponding approximate optimum.

Gradient is defined as a vector of partial derivatives. See Eq. 1-10. Physically, the gradient of a function points in the direction of most rapidly increasing function values.

Gradient-based Numerical Optimizer is any optimizer that uses function gradient information to search for an optimal design. The optimizer in NX Nastran is of this class.

Grid Perturbation refers to a small change in a grid point coordinate for a corresponding infinitesimal design variable change. These small variations are used in differencing schemes for design sensitivity analysis.

Grid Variation refers to a finite change in a grid point coordinate for a corresponding change in a design variable. This results in a finite change of structural shape.

Hard Convergence refers to the design cycle convergence test that compares the results of two consecutive finite element analyses. The terminology is chosen to indicate that the test is based on hard, or conclusive, evidence.

Independent Design Variable is a design variable defined on a DESVAR entry and is not expressed as a function of any other design variables. (DLINK entries are used to define this dependence.) By default, all design variables are independent. In design optimization, the optimizer varies the independent variables directly.

Infeasible Design is defined as a design that violates one or more constraints.

Kuhn-Tucker Conditions provide the formal definition of an optimal design (see Numerical Optimization). These conditions are implicitly evaluated in NX Nastran in connection with convergence detection at the optimizer level.

Move Limits serve to limit the region in which the optimizer may search during the current approximate optimization. Since the approximate subproblem is only valid in the region of the current design (see Approximation Concepts in Design Optimization), move limits serve to restrict the search in this approximate design space to avoid numerical ill-conditioning due to poor approximations.

Nonunique Optimal Design is an optimum design at which the objective function and all active constraints are relatively insensitive to changes in the design variables. See Figure 3-15 in Convergence Tests , for a qualitative explanation of the situation.

Objective Function (Design Objective) is the function that the optimizer seeks to minimize. The objective function must be a continuous function of the design variables.

One-Dimensional Search is the search process at the optimizer level in which all changes to the vector of design variables are characterized by changes in a single-scalar parameter, α. See the definition for Search Direction, and Eq. 1-12.

Optimum Design is defined as a point in the design space for which the objective function is minimized and the Kuhn-Tucker Conditions are satisfied. The optimum might be feasible if all constraints are satisfied or infeasible if one or more constraints are violated. If relative minima exist in the design space, other optimal designs can exist.

Primary Model (or Primary Structure) refers to the finite element model used for analysis. The distinction arises when Auxiliary Models are used for shape optimization. See the definitions for Auxiliary Models and Auxiliary Boundary Models.

Property Optimization defines a design task in which the properties describing the analysis model are allowed to vary. Also frequently referred to as sizing optimization, quantities such as plate thicknesses, bar element moments of inertia, composite ply orientations, and so on, are modified in search of an improved design.

Pseudo-Load Vector refers to the right-hand side vector in the sensitivity analysis equations. See Design Sensitivity Analysis .

Reduced Basis Formulation uses a small number of design variables to describe changes to a large number of analysis model properties or grid coordinates. Essentially, each design variable acts as a multiplier of a (constant) vector. The linear superposition of these so-called basis vectors represents the total design changes. See Eq. 2-23 and Eq. 3-3.

Search Direction (Search Vector) an *n*-dimensional vector, where *n* is the number of independent design variables that characterizes a particular search path used by the optimizer. The optimizer may search along a number of different paths in connection with a given design cycle. Each of these searches is termed a design iteration. Upon completion of these iterations, an approximate optimum is at hand. This approximate optimization process is repeated in subsequent design cycles.

Second Level Responses, also known as synthetic responses, are formulated by the engineer using NX Nastran's equation input capability. These responses are defined using DRESP2 and DEQATN Bulk Data entries.

Semianalytic Sensitivity Analysis refers to an approximate solution of the analytic sensitivity equations. With this approach, the appropriate system equations are differentiated and the structural matrices approximated using finite differences. The resulting approximate set of equations is then explicitly solved for the response derivatives. See Design Sensitivity Analysis , to see how this approach is used for the various analysis disciplines.

Shape Basis Vector is a basis vector used in connection with shape sensitivity and optimization. The vector is a collection of directions along which designed grids can move. Each element of the vector represents the magnitude of change in a grid coordinate due to a unit change in a design variable. See the definition for Basis Vector.

Shape Design Variable is a design variable used to describe structural shape variations. Theoretically, shape design variables act as multipliers of shape basis vectors, but in practice can often be thought of, for example, as a radius, a coordinate location defining the center of a cutout, the percentage addition of taper to a control arm, and so on.

Shape Optimization defines a design task in which boundaries and connection point locations are allowed to change in search of an improved design.

Shape Sensitivity computes the rate of change of structural responses with respect to shape-defining design variables. The grid coordinate changes are expressed in terms of design variable changes using the methods of Relating Design Variables to Shape Changes .

Side Constraint is either an upper or a lower bound on a design variable. Since the optimizer never searches outside of these bounds, these side constraints essentially define the limits of the design space.

Soft Convergence refers to the design cycle convergence test that compares the results of an approximate optimization with those from the previous finite element analysis. This test, while not as conclusive as hard convergence, may be a suitable test to indicate convergence, especially if the associated analysis cost is high.

T**ype-1 Design Variable-to-Property Relations** express analysis model properties as linear functions of design variables. They are defined on DVPREL1 Bulk Data entries.

Type-2 Design Variable-to-Property Relations express analysis model properties in terms of design variables using NX Nastran's equation input capability. They are defined on DVPREL2 Bulk Data entries and use equations defined on DEQATN entries.

Violated Constraint is, strictly speaking, any constraint whose value is greater than zero. In NX Nastran, the constant CTMIN (default = 0.003) is used to provide a numerical zero. Thus, up to one-half of one percent constraint violation is tolerated before a constraint is considered violated. This improves the numerical conditioning of the problem, since trying to precisely satisfy an inequality relation is often a waste of computational resources.

Appendix B: Nomenclature

$\cdot$	Vector dot product
x	Vector cross product
•	Scalar multiplication
$\lvert a \rvert$	Absolute value of the quantity a
$[B\ ]$	Matrix of damping terms in the finite element analysis
$F(\vec{x})$	Objective function
$\tilde{F}(\vec{x})$	Approximate objective function
g_L	Lower-bound constraint
g_U	Upper-bound constraint
$\tilde{g}_{jD}$	Direct variable approximation for the j-th constriant
$\tilde{g}_{jR}$	Reciprocal variable approximation for the j-th constraint
$g_j(\vec{x})$	j-th constraint function
$\tilde{g}_j(\vec{x})$	Approximation of the j-th constraint function
$\{\delta g\}_i$	Variation in the i-th grid coordinates for finite difference sensitivity analysis
$\{\Delta g\}_i$	Coordinate variation in the i-th grid point for shape optimization
$h_k(\vec{x})$	k-th equality constriant

$[I]$	Identity matrix
$[K]$	Stiffness matrix in the finite element analysis
$[K_d]$	Differential stiffness matrix
$[M]$	Mass matrix in the finite element analysis
p^U	Upper move limit on j-th property
p^L	Lower move limit on j-th property
$\vec{p}^{\,0}$	Vector of initial property values
$\{P\}$	Load vector in the finite element analysis
r_j^L	Lower bound on the j-th structural response
r_j^U	Upper bound on the j-th structural response
$r_j^{(1)}$	j-th first-level (DRESP2) response
$r_j^{(2)}$	j-th second-level (DRESP1) response
$\vec{r}$	Vector of analysis model responses (displacements, stresses, eigenvalue, etc.)
$\vec{S}$	Search vector in the optimization’s one-dimensional search
$[T]_{MXN}$	Matrix of constant coefficients in reduced basis formulation where M > N
$\{U\}$	Vector of displacements
$\vec{Y}$	Vector of intermediate variables used in formal approximations
x_i^l	Lower-bound side constraint on i-th design variable
x_i^u	Upper-bound side constraint on i-th design variable

$\vec{x}$	Vector of design variables
$\vec{x}^0$	Initial value of vector of design variables (this characterizes the initial design).
$\vec{x}^*$	Vector of design variables at the optimal design
$\vec{x}_D$	Vector of dependent design variables
$\vec{x}_I$	Vector of independent design variables
$\{\ \}^T$	Transpose of a vector
$[\ ]^T$	Transpose of a matrix
α	One-dimensional search parameter
α^*	Optimal value of one-dimensional search parameter for the current search direction $\vec{S}$
$\frac{\partial}{\partial x_i}, \frac{\delta}{\delta x_i}$	Partial derivatives with respect to the i-th design variable
Δ	Finite difference move, delta
λ_i	Lagrange multiplier used in the Kuhn-Tucker conditions
λ_{ij}	Sensitivity coefficient for the i-th design variable and j-th response
∇	Gradient operator
$\{\varphi_n\}$	n-th eigenvector
$[\Phi]$	Modal transformation matrix
ρ	Mass density
ω	Frequency of oscillation of a dynamic system (rad/s)

$\{\xi\}$	Vector of modal responses

Appendix C: Commonly Used Commands for Design Optimization

C.1 Case Control Command

This section lists commonly used Case Control commands for optimization. See the *NX Nastran Quick Reference Guide* for complete descriptions of these entries.

- DRSPAN

C.2 Bulk Data Entries

This section lists commonly used Bulk Data entries for optimization. See the *NX Nastran Quick Reference Guide* for complete descriptions of these entries.

- DCONADD
- DCONSTR
- DDVAL
- DEQATN
- DESVAR
- DLINK
- DOPTPRM
- DRESP1
- DRESP2
- DRESP3
- DTI,DFRFNC
- DSCREEN
- DTABLE
- DVBSHAP
- DVCREL1
- DVCREL2

- DVGEOM
- DVGRID
- DVMREL1
- DVMREL2
- DVPREL1
- DVPREL2
- DVSHAP

C.3 Parameters

This section lists commonly used parameters for optimization. See the *NX Nastran Quick Reference Guide* for complete descriptions of these parameters.

- AUTOADJ
- CDIF
- DESPCH
- DSNOKD
- DESPCH1
- DPEPS
- DSZERO
- NASPRT
- OPTEXIT
- SENSUOO
- SOFTEXIT
- UPDTBSH

Appendix D: Numerical Optimization

D.1 Introduction

The basic features of numerical optimizers were introduced in Getting Started . This general overview is probably sufficient in most situations to let you follow the tasks performed by the optimizer, to understand how to pose the design problem for effective solution, and to interpret most of the optimization-related output.

This section completes the discussion begun in Getting Started and is intended for the intermediate- to advanced-level reader who wishes to know more about the implementation of the optimizer used in NX Nastran. This section has largely been excerpted from the DOT User's Manual[1]. In addition, Vanderplaats' *Numerical Optimization Techniques for Engineering Design with Applications*[2] is recommended further reading on this topic, because it includes theoretical details that are beyond the scope of this section.

The processes described here are a part of the code and are not subject to change by the engineer. However, many of those methods use numerical constants in connection with convergence detection, one-dimensional search control, and so on. Defaults are provided for all of these internal optimizer parameters, but these may be overridden using the DOPTPRM Bulk Data entry. This section presents the advanced-level information necessary to make these override decisions.

Chapter 1 introduced the Basic Optimization Problem Statement. Recall that this provides the definition of the design variables, objective, and constraints as

$$\text{minimize } F(\vec{x}) \text{ objective}$$

Equation D-1.

$$\text{subject to } g_j(\vec{x}) \leq 0 \quad \text{j} = 1, \ldots, n_g \quad \text{inequality constraints}$$

Equation D-2.

$$h_k(\vec{x}) = 0 \quad \text{k} = 1, \ldots, n_h \qquad \text{equality constraints}$$

Equation D-3.

1. DOT User's Manual, Vanderplaats, Miura and Associates, Inc., 1990.
2. Vanderplaats, G. N. *Numerical Optimization Techniques for Engineering Design with Applications*, McGraw Hill, 1984.

$$x_i^l \le x_i \le x_i^u \quad i = 1, \ldots, n \qquad \text{side constraints}$$

Equation D-4.

where:

$$\vec{x} = \{x_1, x_2, \ldots, x_n\} \qquad \text{design variables}$$

Equation D-5.

It is within the design space defined by the above problem statement that the optimizer searches for a best design.

Chapter 1 also introduced the analogy of a blindfolded search along the side of a hill for the point of lowest elevation, to describe some of the basic processes used by gradient-based numerical optimizers. At that time, we introduced the idea of a one-dimensional search and the steepest descent algorithm as a particular method of establishing this search direction. However, we also noted that this method, in addition to not being particularly efficient, cannot address those situations in which one or more constraints were active. In this section, we discuss how these issues are resolved by the optimizer in NX Nastran and we then move on to consider some of the other details of the search process.

Recall that the design variable vector update can be written as

$$\vec{x}^1 = \vec{x}^0 + \alpha^* \vec{S}^1$$

Equation D-6.

where $\vec{x}^0$ is the initial vector of design variables, $\vec{S}^1$ is the search vector, and α is the search parameter. Eq. D-6 represents a one-dimensional search, since the update on $\vec{x}^1$ depends only on the single scalar parameter α. α^* is the value of α that yields the optimal design in the direction defined by $\vec{S}$. In other words, to continue with the hill analogy, we have either encountered a fence or are at the bottom of a valley in the direction of $\vec{S}^1$. Finding α^* completes the first iteration in the “design” process.

If we were at the bottom of a valley, we could just repeat the process of finding a new steepest descent direction and keep moving. In practice, there is a better choice of direction called a conjugate direction, which we will discuss shortly. However, regardless of the search direction used, the concept is the same. We find a search direction down the hill and move in that direction as far as possible.

Now, assume instead that we (or the optimizer) have encountered a fence. In order to make any further improvement in the “design,” a new search direction $\vec{S}^2$ must be found that continues to reduce our elevation on the hill but keeps us inside of the fence. Here we seek a “usable-feasible” direction, in which a usable direction is one that moves us downhill and a feasible direction is one that keeps us inside of the fence. This situation is shown in Figure D-1. The mathematical definition of a usable search direction is

$$\nabla F(\vec{x}) \cdot \vec{S} \leq 0$$

Equation D-7.

Eq. D-7 is just the scalar product (dot product) of the gradient of the objective function with the search direction. The dot product is the magnitude of $\nabla F(\vec{x})$ times the magnitude of $\vec{S}$ times the cosine of the angle between the two vectors. Thus, the cosine of the angle determines the sign of the product, since the magnitudes are positive numbers. For the cosine to be zero or negative, the angle between the vectors must be between 90 and 270 degrees. If the cosine is zero, the search direction is at an angle of 90 or 270 degrees from the gradient of the objective function. A move in this direction follows a contour on the hill and (for a small move) does not increase or decrease the function. If the cosine is –1.0, the direction is opposite to the direction of $\nabla F(\vec{x})$ and is the direction of steepest descent. Thus, we wish to find a search direction that makes the left-hand side of Eq. D-7 as negative as possible. However, this direction must remain within a critical constraint (fence). This is the feasibility requirement that is similar to the usability requirement but now is stated with respect to the constraint

$$\nabla g_j(\vec{x}) \cdot \vec{S} \leq 0$$

Equation D-8.

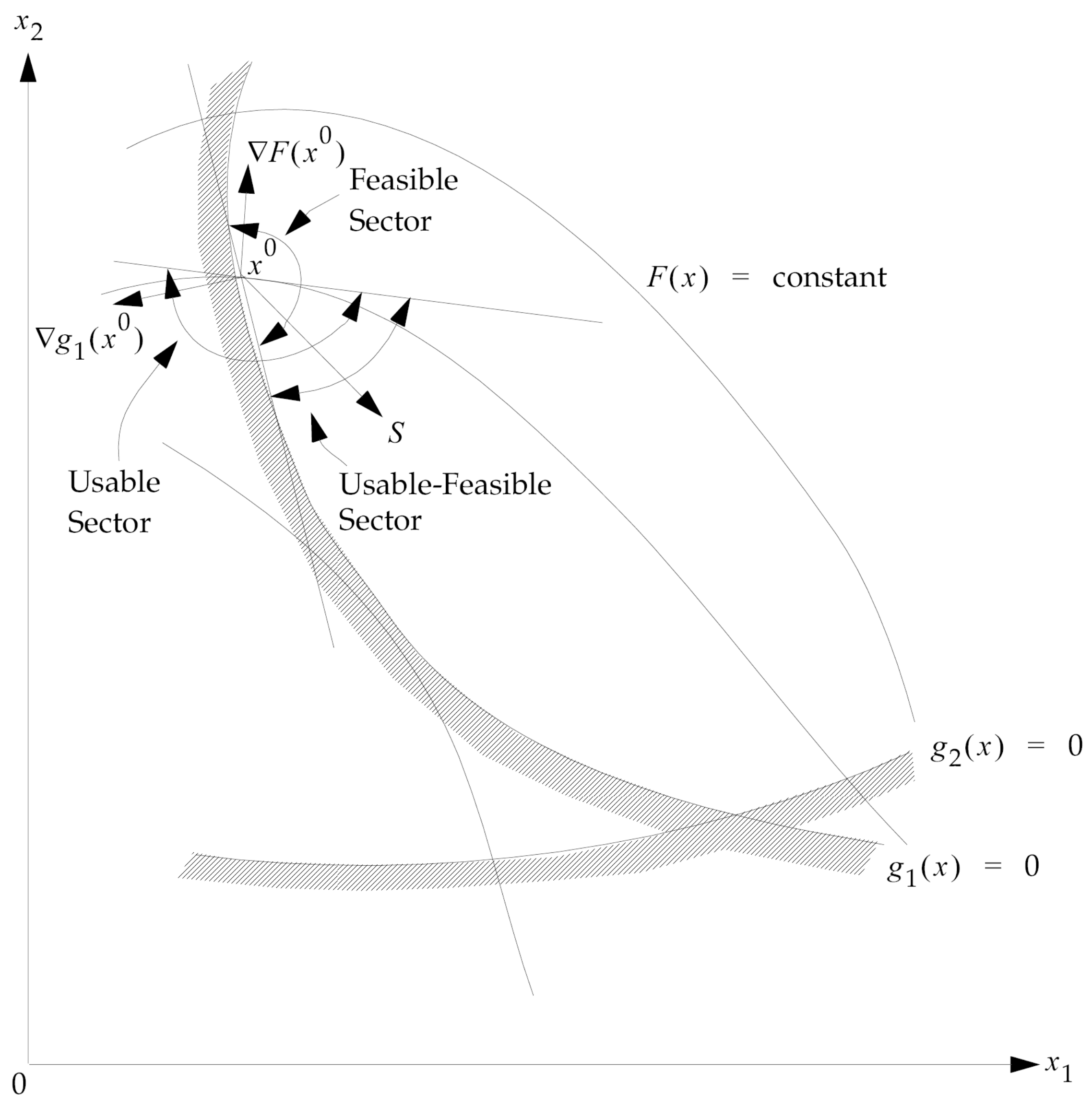


Figure D-1. Usable-Feasible Search Direction

Just as for the objective function, the angle between the search direction $\vec{S}$ and the gradient of the constraint must be between 90 and 270 degrees. If the angle is exactly 90 or 270 degrees, the search direction is tangent to the constraint boundary (fence). To find the search direction that makes the greatest possible improvement in the objective function but still follows or moves inside of the fences, we combine the usability and feasibility requirements. This combination creates the following suboptimization task: Find the components of the search direction $\vec{S}$ in order to

$$\text{minimize } \nabla F(\vec{x}) \cdot (\vec{S})$$

Equation D-9.

$$\text{subject to: } \nabla g_j(\vec{x}) \cdot \vec{S} \leq 0 \qquad j \in J$$

Equation D-10.

$$\vec{S} \cdot \vec{S} \leq 1$$

Equation D-11.

where *J* is the set of constraints whose values are zero within some numerical tolerance. This is the set of active constraints. The purpose of Eq. D-11 is simply to prevent an unbounded solution to the problem defined by Eq. D-9 and Eq. D-10. In the case of a simple two-variable problem, finding the appropriate search direction is quite easy and may be done graphically. In the more general case where there are numerous design variables as well as several active constraints, this becomes a subproblem that is solved as part of the optimization. This problem is linear in $\vec{S}$ except for the quadratic constraint of Eq. D-11. Details regarding the solution of this subproblem can be found in *Numerical Optimization Techniques for Engineering Design with Applications*[2].

Assuming we can find a usable-feasible search direction, we can now search in this direction until we can make no further improvement. The subproblem of finding a new usable-feasible search direction is repeated and continues until no search direction can be found that improves the design without violating one or more constraints. We call this point the “optimum.” In the present example, we assumed the initial design was feasible. In practice, we often start outside of one or more fences, or from an initially infeasible design.

If this is the case, the optimizer’s first task is to search for a feasible design. Frequently, the objective function must be increased in this process. For example, it may be necessary to add structural mass in order to overcome element stress constraints. However, the optimizer in NX Nastran will search for a feasible region, allowing for the smallest possible increases in the objective. Having found the feasible domain (if one exists), it can then proceed with minimizing the objective.

The question now arises: How do we know that we have reached the optimum? The answer can be found in what are known as the Kuhn-Tucker conditions[3] [4]. In the case of an unconstrained problem (omitting Eq. D-2 through Eq. D-4), this is simply the condition where the objective function vanishes (i.e., equals zero). In the case of the constrained optimization problem considered here, the conditions of optimality are more complex. In this case, the governing equation is the stationary condition of the Lagrangian function:

$$L(\vec{x}, \vec{\lambda}) = F(\vec{x}) + \sum_{j=1}^{M} \lambda_j g_j(\vec{x})$$

Equation D-12.

where

3. Vanderplaats, G. N., “An Efficient Feasible Direction Algorithm for Design Synthesis, “ *AIAA Journal*, Vol. 22 No. 11, Nov. 1984,
4. Zangwill, W. I. *Nonlinear Programming: A Unified Approach*, Prentice-Hall, Englewood Cliffs, NJ,, 1969.

$$\lambda_j \geq 0$$

Equation D-13.

The Kuhn-Tucker conditions dictate that the Lagrangian function $L(\vec{x}, \vec{\lambda})$ must have a vanishing gradient at the optimum design, x^*. However, we must also remember the original optimization problem and the inequality condition of Eq. D-2. In addition, we note that Eq. D-3 for equality constraints is restated in NX Nastran as a set of inequality constraints. When all of these conditions are considered, they lead to the statement of the Kuhn-Tucker necessary conditions for optimality:

$$\text{Condition 1:} \qquad \vec{x}^* \text{ is feasible. Therefore, for all } j,\ g_j(\vec{x}^*) \leq 0$$

Equation D-14.

$$\text{Condition 2:} \qquad \lambda_j g_j(\vec{x}^*) = 0 \ \ \text{(the product of } \lambda_j \text{ and } g_j(\vec{x}^*) \text{ equals zero)}$$

Equation D-15.

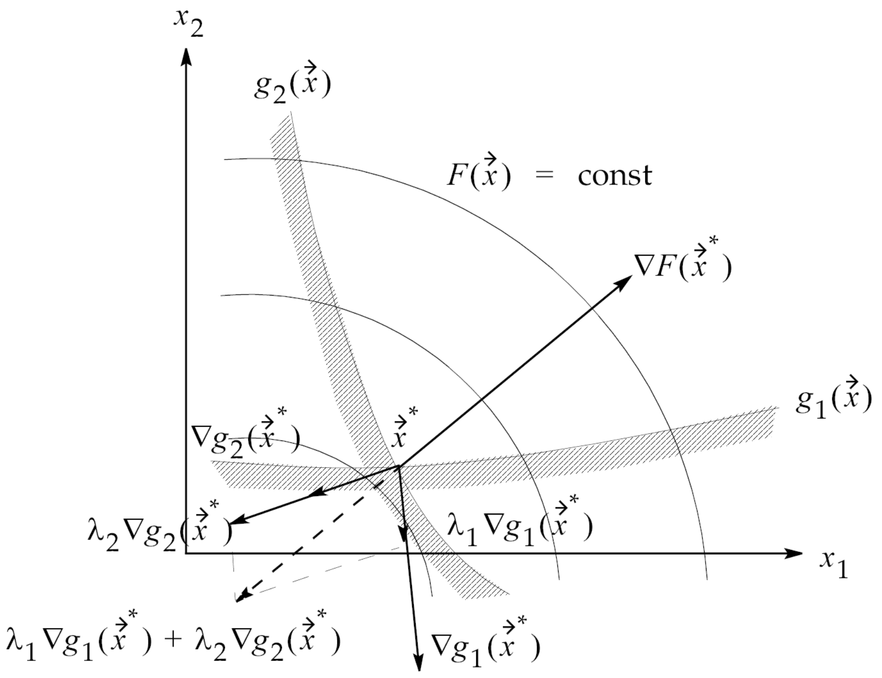
$$\text{Condition 3:} \qquad \nabla F(\vec{x}^*) + \sum_{j=1}^{M} \lambda_j \nabla g_j(\vec{x}^*) = 0$$

Equation D-16.

$$\lambda_j \geq 0 \qquad j = 1, 2, \ldots, M$$

Equation D-17.

The physical interpretation of these conditions is that the sum of the gradient of the objective and the scalars λ_j times the associated gradients of all active constraints must vectorally add to zero. This is much like the statement for static equilibrium where the sum of all internal and external forces at any given point must vectorally add to zero. Figure D-2 shows this situation for a simple two-variable function space with two active constraints at the optimum. Eq. D-15 states that the corresponding Lagrange multiplier is zero for all non-active constraints.



x_2
$g_2(\vec{x})$
$F(\vec{x}) = \text{const}$
$\nabla F(\vec{x}^*)$
$g_1(\vec{x})$
$\nabla g_2(\vec{x}^*)$
$\vec{x}^*$
$\lambda_2 \nabla g_2(\vec{x}^*)$
$\lambda_1 \nabla g_1(\vec{x}^*)$
x_1
$\lambda_1 \nabla g_1(\vec{x}^*) + \lambda_2 \nabla g_2(\vec{x}^*)$
$\nabla g_1(\vec{x}^*)$

Figure D-2. Kuhn-Tucker Conditions at a Constrained Optimum

These are only necessary conditions and the definition here is actually a bit more restrictive than required in some cases. However, it does provide the essential idea. In practice, it is difficult to numerically find a design that precisely satisfies the Kuhn-Tucker conditions. Also, numerous designs might actually satisfy these conditions since there may be more than one constrained minimum. The importance of the Kuhn-Tucker conditions is that an understanding of the necessary conditions for optimality gives us some knowledge of what is needed to achieve an optimum.

The simple example of descending a hill by the approach outlined above is the physical interpretation of a class of optimization methods known as feasible direction techniques. A multitude of other numerical search algorithms is available to solve the general optimization problem (see Vanderplaats, “An Efficient Feasible Direction Algorithm For Design Synthesis,”[3]). Most of the more powerful methods update the design using the aforementioned relation:

$$\vec{x}^{q} = \vec{x}^{q-1} + \alpha^* \vec{S}^{q}$$

Equation D-18.

where:

q	=	iteration number
$\vec{S}$	=	vector search direction
α^*	=	scalar move parameter

Eq. D-18 is the same as Eq. D-6 except that we have replaced the superscripts with the iteration counter q. Just as in the example of searching down the hill, the product $\alpha^* \vec{S}$ is the design modification at the current step. Note that this is very similar to the traditional engineering approach of modifying a current best design instead of using a completely new design. An initial design $\vec{x}^0$ must be provided, but it need not satisfy all of the constraints. (Indeed, one of the most powerful uses of optimization is to find a feasible solution to a complicated problem).

To determine a search direction $\vec{S}$ that improves the design, gradients of the objective and critical constraints must be supplied. Ideally, these are computed analytically or semi-analytically as is done in NX Nastran. This process dramatically increases the size of the problem that can be efficiently solved. Finally, a “one-dimensional” search is performed by trying several values of α and interpolating for the one that gives a minimum objective while satisfying the constraints. During this process, the objective and the constraints must be repeatedly evaluated. Here the use of approximation methods plays a major role because the evaluation of these constraints would otherwise require a full finite element analysis. In NX Nastran, we only evaluate the approximate functions, which is a relatively inexpensive process.

In the following sections we will outline the overall optimization process and identify the steps taken to reach the optimum. The presentation here will be necessarily brief, and the reader is referred to Vanderplaats, *Numerical Optimization Techniques for Engineering Design with Applications*[2] for more details.

D.2 The Modified Feasible Direction Algorithm

Now we turn to the actual task of solving the approximate problem. The method described here is referred to as the Modified Method of Feasible Directions (see 4). At this point it is assumed that we are provided with an objective function $F(\vec{x})$ and constraints $g_j(x) \leq 0$, $j = 1, 2, ..., n_g$ as well as lower and upper bounds on the design variables. Also, the gradients of the objective and constraints are available. Thus, we are solving the general problem defined by Eq. 1-1, 1-2, and 1-4, which are repeated here for convenience:

Find the set of design variables x_i, $i = 1, 2, ..., n$ that

$$\text{minimize } F(\vec{x})$$

Equation D-19.

$$\text{subject to } g_j(\vec{x}) \leq 0 \qquad j = 1, 2, \ldots, n_g$$

Equation D-20.

$$x_i^L \leq x_i \leq x_i^U \qquad i = 1, 2, \ldots, n$$

Equation D-21.

At this point the optimizer does not know what kind of a problem it is solving. It is simply minimizing a function subject to inequality constraints.

Given an initial x-vector x^o, the design will be updated according to Eq. D-18, which is also repeated here

$$\vec{x}^{q} = \vec{x}^{q-1} + \alpha^{*}\vec{S}^{q}$$

Equation D-22.

The overall optimization process now proceeds in the following steps:

1. Start, $q = 0, \vec{x} = \vec{x}$.
2. *q* =*q* + 1.
3. Evaluate $F(\vec{x})$ and $g_j(\vec{x})$ where *J* = 1, 2, ..., n_g
4. Identify the set of critical and near critical constraints *j* Є *J*.
5. Calculate $\nabla F(\vec{x})$ and $\nabla g_j(\vec{x})$ for all *j* Є*J*.
6. Determine a usable-feasible search direction $\vec{S}^{q}$.
7. Perform a one-dimensional search to find α^{*} .
8. Set $\vec{x}^{q} = \vec{x}^{q-1} + \alpha^{*}\vec{S}^{q}$.
9. Check for convergence to the optimum. If satisfied, exit. Otherwise, go to Step 2.

The critical parts of the optimization task consist of the following:

1. Finding a usable-feasible search direction, $\vec{S}^{*}$
2. Finding the scalar parameter α^{*} that will minimize $F(\vec{x}^{q-1} + x^{*}\vec{S}^{q})$ subject to the constraints.
3. Testing for convergence to the optimum $\vec{x}^{*}$, and terminating if convergence is achieved.

We will discuss each of these in turn.

D.3 $\vec{S}^{q}$ Finding the Search Direction

The first step in finding the search direction is to determine which constraints, if any, are active or violated. Here an active constraint is defined as one with a value between CT and CTMIN, where CT is a small negative number and CTMIN is a small positive number. Remember that constraints must be negative to be feasible; therefore, if $g_j(\vec{x})$ is less (e.g., more negative) than CT, it is not considered active. Also, when we approach a constraint boundary, it makes sense to begin “pushing” away from that “fence.” Therefore, we initially choose a relatively large value for CT, such as –0.03 (i.e., within three percent of being mathematically active).

On the other hand, any constraint with a positive value is mathematically violated. However, trying to achieve a precise zero on the computer is not meaningful. Also, loads, material properties, etc., are not really known precisely. Furthermore, the responses calculated by the finite element analysis are only approximate because of the nature of the method. Therefore, we allow a small positive constraint value before identifying a constraint as violated. This is the value of CTMIN typically taken as 0.003 (three-tenths of one percent violation). Thus, the governing definitions are

$$g_j(x) < \mathrm{CT} \qquad \text{Inactive}$$

Equation D-23.

$$\mathrm{CT} \le g_j(x) \le \mathrm{CTMIN} \qquad \text{Active}$$

Equation D-24.

$$g_j(x) > \mathrm{CTMIN} \qquad \text{Violated}$$

Equation D-25.

These tolerances are shown in Figure D-3 for a single constraint in a two design variable space.

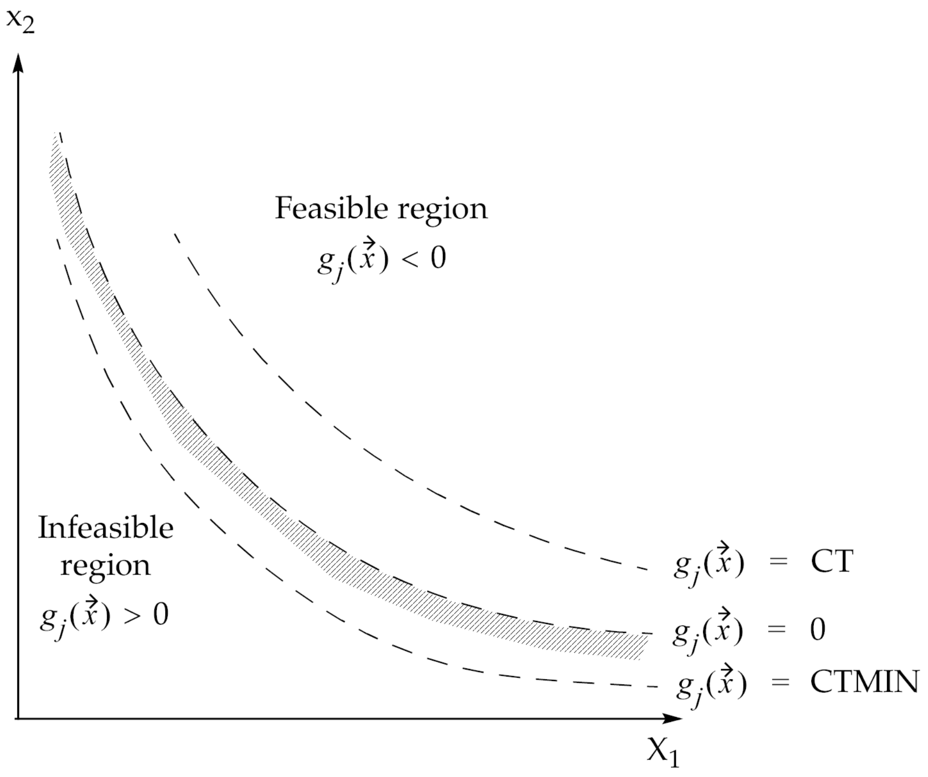


Figure D-3. CT and CTMIN

The use of Eq. D-23 through Eq. D-25 underscores the importance of normalizing constraints. In NX Nastran, this is done automatically using the constraint bounds as normalizing factors. However, certain precautions should be taken by the engineer when formulating second-level responses as discussed in Design Modeling for Sensitivity and Optimization .

Using the active constraint criteria, the algorithm first sorts all the constraints and identifies those that are active or violated. Then, the gradients of the objective function and all the active and violated constraints are calculated. Thereafter, a usable-feasible search direction is found (if one exists). In this case, there are three possibilities:

1. There are no active or violated constraints.
2. There are active constraints but no violated constraints.
3. There are one or more violated constraints.

Each of these possibilities is handled differently.

No Active or Violated Constraints (Unconstrained Minimization)

Frequently, at the beginning of the optimization process there are no active or violated constraints. In this case, the feasibility requirement is automatically met because we can move in any direction for at least a short distance without violating any constraints. Thus, we only need to find a usable direction, which is the one that points downhill. It does not have to be the steepest descent direction, but to start the process, this is the preferred choice. Therefore, the initial search direction is simply:

$$\vec{S}^q = -\nabla F(\vec{x}^{q-1})$$

Equation D-26.

The steepest descent direction is only used if this is the beginning of the optimization (q = 1) or if the results of the last iteration yielded no active or violated constraints.

Now assume the last search direction is in the steepest descent direction and there are still no active constraints. (If there were no violated constraints before, there will not be any now since the optimizer would have stopped the search before any violation.) We could again search in the steepest descent direction, and this is commonly done. Such a direction would be perpendicular to the previous direction. However, there are very solid theoretical reasons not to do this, and experience attests to the accuracy of the theory. Instead, we move in what is called a “conjugate” search direction, or more precisely, an A-conjugate direction, where A is the matrix of second partial derivatives of the objective function. The A matrix is not actually computed, but there are methods for approximating A that offer a guaranteed convergence rate for problems where the objective is a quadratic function. (In fact, the successive use of steepest descent directions may never converge.)

While there are methods that are considered more powerful than the one used here, the simple conjugate direction method has proved to be quite reliable. The Fletcher-Reeves conjugate direction method[5] used here, defines a search direction as:

5. Fletcher, R., and Reeves, C. M., “Function Minimization by Conjugate Gradients,” Computer J., Vol. 7, No. 2, pp. 149–154, 1964.

$$\vec{S}^{q} = -\nabla F(\vec{x}^{q-1}) + \beta \vec{S}^{q-1}$$

Equation D-27.

$$\text{where } \beta = \frac{\left\| \nabla F(\vec{x}^{q-1}) \right\|^2}{\left\| \nabla F(\vec{x}^{q-2}) \right\|^2}$$

Equation D-28.

with the double vertical bars indicating the Euclidean norm of each gradient.

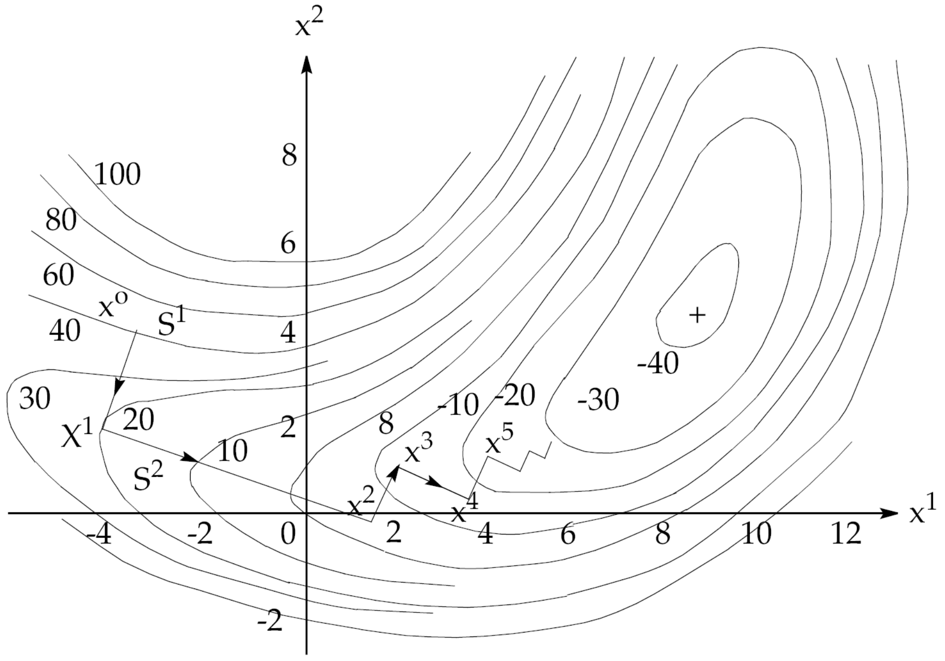


Figure D-4. Geometric Interpretation of the Steepest Descent Method

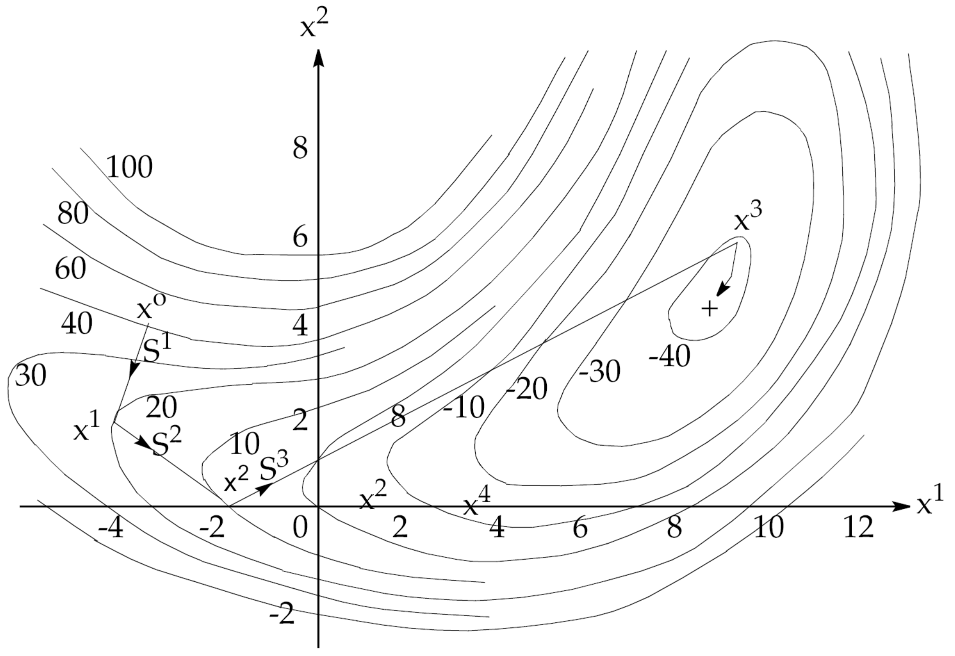


Figure D-5. Geometric Interpretation of the Fletcher-Reeves Method

The advantage of using the conjugate search direction is seen from Figure D-4 and Figure D-5, which show a simple two-variable design space. In Figure D-4 the search directions are the steepest descent directions, whereas in Figure D-5 the conjugate directions are used. Note that in Figure D-4, the search direction is always perpendicular to the previous direction. On the other hand, in Figure D-5, each search direction uses the steepest descent direction plus some fraction of the previous direction. This makes physical sense. If progress is made in direction S^1, it is reasonable to bias direction S^2 a bit in that direction rather than simply using a steepest descent direction. This is exactly what Eq. D-27 and Eq. D-28 accomplish.

This algorithm is extremely simple but, as seen from Figure D-4 and Figure D-5, it dramatically improves the rate of convergence to the optimum. As noted earlier, there are other methods (called variable metric methods) that are considered better than the one just described. In fact, NX Nastran uses the BFGS (Broydon, Fletcher, Goldfarb, and Shanno) method[6], [7], [8], [9]. Also, the geometrical interpretation is almost exactly the same as the Fletcher-Reeves algorithm. Therefore, there is no loss in the physical interpretation of the methods. However, the details of the method are beyond the scope of this discussion. A key issue to note is that the steepest descent method, which even today is published as a breakthrough in engineering optimization, is generally a poor method that should not be used in a modern design environment.

Active But No Violated Constraints

The more common search direction problem, in which the initial design is feasible and there is at least one active constraint, is to find a search direction that improves the design but moves parallel to or

6. Broyden, C. G., “The Convergence of a Class of Double Rank Minimization Algorithms, Parts I and II,” J. Inst. Math. Appl., Vol. 6, pp 76–90, 222–231, 1970.,
7. Fletcher, R., “A New Approach to Variable Metric Algorithms,” Computer J., SIAM, Vol. 13, pp. 317–322, 1970.,
8. Goldfarb, D. “A Family of Variable Metric methods Derived by Variational Means,” Math. Comput., Vol. 24, pp. 23–26, 1970.,
9. Shanno, D. F., “Conditioning of Quasi-Newton Methods for Function Minimization,” Math., Comput., Vol. 24, pp. 647–656, 1970

away from the constraint. To solve this problem, a search direction $\vec{S}$ must first be found that reduces the objective function without violating any currently active constraints. The following equations state the problem in mathematical terms:

Find the search direction $\vec{S}^q$ that

$$\text{minimize}\, \nabla F(\vec{x}^{\,q-1}) \cdot \vec{S}^q$$

Equation D-29.

$$\text{subject to}\, \nabla g_j(\vec{x}^{\,q-1}) \cdot \vec{S}^q \qquad j \in J \qquad \text{Feasibility Condition}$$

Equation D-30.

$$\vec{S}^q \cdot \vec{S}^q \le 1 \qquad \text{Bounds on } S$$

Equation D-31.

This is the same problem as shown in Eq. D-9 through Eq. D-11 for finding a usable-feasible search direction in the physical example of searching for the lowest point on the hill. Note that the scalar product is the magnitude of the two vectors times the cosine of the angle between them. Thus, the objective of this suboptimization problem is

$$\left\|\nabla F(\vec{x}^{\,q-1})\right\| \left\|\vec{S}\right\| \cos(\theta)$$

Equation D-32.

Since we are minimizing this function, we want the cosine of the angle between the two vectors to be as large a negative number as possible but within the restriction of Eq. D-30. Alternatively, for any angle θ between 90 and 270 degrees, Eq. D-32 can be made more negative if the magnitude of $\vec{S}$ is increased. Also, if $\vec{S}$ satisfies the requirements of Eq. D-30, then any increase in the magnitude of $\vec{S}$ also satisfies this equation. Therefore, it is necessary to bound $\vec{S}$, which is accomplished using Eq. D-31.

Assuming the resulting objective function from this subproblem is negative, the usability requirement of Eq. D-7 is satisfied. If the objective function (defined by Eq. D-29) cannot be forced to be negative, then it follows that no direction exists that reduces the objective function while staying inside of the constraints. If this is the case, the Kuhn-Tucker conditions are satisfied, and the optimization process can be terminated.

In practice, the decision is not quite this simple because Eq. D-30 may include constraints that are within a tolerance of CT but are not really critical. Therefore, we reduce CT and delete any constraints from the active set J that are not within the new tolerance. We then solve this direction finding problem again to determine if a more precise satisfaction of the constraints can be achieved. If the value of CT is reduced in magnitude to CTMIN (CT is negative) and no negative objective is found for the direction finding problem, then this design is judged to have satisfied the Kuhn-Tucker conditions, and the optimization process is stopped.

The next question is: How do we solve this sub-problem of finding the usable-feasible search direction $\vec{S}$? Unfortunately, the details are beyond the scope of this discussion. For our purposes, it is sufficient to note that the problem can be converted to an iterative form of solving a set of simultaneous equations with some conditions on the variables. The actual algorithm is similar to that used in traditional linear programming. The details of this suboptimization problem are given in Vanderplaats, “An Efficient Feasible Direction Algorithm for Design Synthesis,”[3].

One or More Violated Constraints

Now consider the case where one or more constraints are violated. Such a case is shown in Figure D-6 where constraint $g_1(x)$ is violated and $g_2(x)$ is active. Now we must find a search direction back toward the feasible region even if it is necessary to increase the objective function to do so. To achieve this, we augment our direction-finding problem of Eq. D-29 through Eq. D-31 with a new variable *W*. This process has no direct physical significance to the problem except as a measure of the constraint violation.

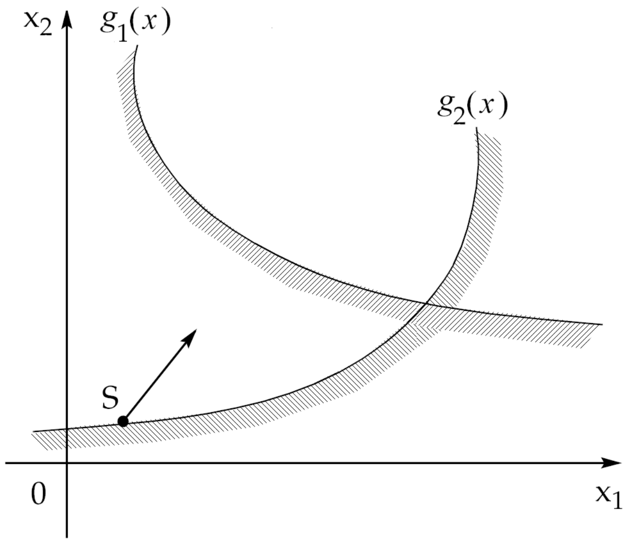


Figure D-6. Violation of Constraint(s)

The new direction finding problem is now:

Find the search direction $\vec{S}$ and artificial variable *W* that will

$$\text{minimize} \quad \nabla F(\vec{x}^{\,q-1}) \cdot \vec{S}^{\,q} - \psi W$$

Equation D-33.

$$\text{subject to} \qquad \nabla g_j(\vec{x}^{\,q-1}) \cdot \vec{S}^{\,q} + \theta_j W \le 0 \qquad j \in J$$

Equation D-34.

$$\vec{S}^{\,q} \cdot \vec{S}^{\,q} + W \le 1$$

Equation D-35.

For discussion, assume that the parameter θ_j in Eq. D-34 is equal to 1.0 and that the parameter ψ is a very large positive number. Then the second term dominates the minimizing of the function defined by Eq. D-33 so any increase in the variable *W* will drive the objective more and more negative (i.e., reduce the objective of the direction finding problem). However, for *W* to increase, the first term in Eq. D-34 must become more and more negative. Since $\vec{S}^{\,q}$ is bounded by Eq. D-35, the cosine of the angle between $\nabla g_j(\vec{x}^{\,q-1})$ and $\vec{S}^{\,q}$ must be moved closer and closer to –1.0. For this to happen, $\vec{S}^{\,q}$ must point in a direction opposite to $\nabla g_j(\vec{x}^{\,q-1})$. That is, $\vec{S}^{\,q}$ must point straight back toward the feasible region. The first term in Eq. D-33 is included simply as a means to reduce the true objective function if possible while seeking a feasible design.

Now, consider the value of θ_j in Eq. D-34. This is referred to as a push-off factor since its value determines how hard to push away from this violated constraint. If θ_j = 0.0, even increasing *W* will not require a move anywhere except tangent to the constraint, and this move will probably not bring the design back to the feasible region. Therefore, some positive value is needed. In the optimizer, the more the constraint is violated, the greater the push-off factor value should be.

This can be accomplished by expressing the push-off factor θ_j as a quadratic function of the j-th constraint such that θ_j = 0 at $g_j(\vec{x}^{\,q-1}) = \mathrm{CT}$. Thus,

$$\theta_j = \theta_0 \left(1.0 - \frac{g_j \vec{x}^{\,q-1}}{\mathrm{CT}} \right)$$

Equation D-36.

$$\theta_j \le 50.0$$

Equation D-37.

Therefore, when a constraint becomes active, it is included in the direction-finding process. However, as the constraint becomes more critical, particularly as it becomes violated, the push-off factor increases. The value of θ_j has been limited to 50.0 as an arbitrary numerical limit. This limit is based on experience that shows this value to be large enough to force the design back to the feasible region without causing too much numerical ill-conditioning in the algorithm.

Finally, consider the value of ψ. This parameter is initially chosen as a relatively small number, such as 5.0. Also, we always normalize $\nabla F(\vec{x}^{\,q-1})$ and $\nabla g_j(\vec{x}^{\,q-1})$ so that the first term in Eq. D-33 and Eq. D-34 is near unity. Thus, the second term in Eq. D-33 dominates, but not too strongly, allowing for the possibility of reducing the objective function while overcoming the constraint violation. If this iteration does not overcome the violation, ψ is increased by a factor of 10, although limited to an upper bound of about 1000, again to avoid numerical ill-conditioning. Usually, this has the result of bringing the design back to the feasible region in a few iterations. If 20 iterations pass without overcoming the constraint violations, the optimization process is terminated on the assumption that no feasible solution exists.

The One-Dimensional Search

Having determined a usable-feasible search direction, the problem now becomes one of determining how far the design can be moved in that direction. Again, a variety of possibilities exist depending on the starting point $\vec{x}^{\,q-1}$. However, in each case, polynomial approximations are used for the objective and constraint functions in the one-dimensional search direction, $\vec{S}$.

The basic concept is to try some initial value for a* in Eq. D-22 and evaluate the corresponding objective and constraint functions. Therefore, the first question to ask is: What is a good first estimate for α^*? At the beginning of the optimization process, very little information is available except the function values and their derivatives with respect to α^*. In fact, at first glance, it is not obvious that the derivatives of the objective and constraints will respect to α^* are available. However, consider the objective function (the same algebra applies to constraints) and create a first order Maclaurin series approximation to $F(\alpha^*)$ in terms of α^*. Remember that

$$F(\vec{x}^{\,q}) = F(\vec{x}^{\,q-1} + \alpha^* \vec{S}^{\,q})$$

Equation D-38.

Thus, a first-order approximation to $F(\vec{x}^{\,q})$ is

$$F(\vec{x}^{\,q}) = F(\vec{x}^{\,q-1}) + \left\{ \sum_{i=1}^{N} \left(\frac{\partial F(\vec{x}^{\,q-1})}{\partial x_i} \right) \bullet \left(\frac{\partial x_i}{\partial \alpha^*} \right) \right\} \bullet \alpha^*$$

Equation D-39.

or

$$F(\vec{x}^{\,q-1}) = F(\vec{x}^{\,q-1}) + \left(\frac{dF(\vec{x}^{\,q-1})}{d\alpha^*} \right) \bullet \alpha^*$$

Equation D-40.

But $\left(\frac{\partial F(\vec{x}^{q-1})}{\partial x_i}\right)$ is just the *j*-th entry of the vector $\nabla F(\vec{x}^{\,q-1})$. Also, from Eq. D-22, $\frac{\partial x_i}{\partial \alpha^*} = s_i$, where s_i is the *i*th entry of the search direction vector. Therefore,

$$\frac{dF(\vec{x}^{\,q-1})}{d\alpha^*} = \nabla F(\vec{x}^{\,q-1}) \bullet \vec{S}^q$$

Equation D-41.

Since both terms in Eq. D-39 are available, we have the slope of the function at $\alpha^* = 0$ for any function (objective or constraint) for which the gradient is available.

Now consider how this information might be used. Since this is the first step in the optimization process, we may optimistically expect to reduce the objective function of our minimization problem by some fraction, for example, 10%, which can be stated as

$$\left(F(\vec{x}^{\,q-1})\right) = F(\vec{x}^{\,q-1}) + \frac{dF(\vec{x}^{\,q-1})}{d\alpha^*} \bullet \alpha^* = F(\vec{x}^{\,q-1}) - 0.1 \bullet \left|F(\vec{x}^{\,q-1})\right|$$

Equation D-42.

from which a proposed α^* is:

$$\alpha^*_{est} = -\frac{0.1 \bullet \left|F(\vec{x}^{\,q-1})\right|}{\left(\frac{dF(\vec{x}^{\,q-1})}{d\alpha^*}\right)}$$

Equation D-43.

This results in an estimate of α^* that reduces the objective function by 10%. However, since the gradients of some constraints are probably available, other possible moves can be calculated as well. Remember that Eq. D-40 applies to a constraint by simply substituting the constraint gradient for the objective gradient. Now, assume that some constraint gradients exist for constraints that are not critical, and it is desired to estimate how far to move to make one of them critical. That is, instead of reducing the value of the function by 10% as was done for the objective, it is desired to estimate a move in α^* that places it near the constraint boundary. Thus, a linear approximation to find $g_j(\vec{x}^{\,q}) = 0.0$ is

$$\left(g_j(\vec{x}^{q}) = \right) \quad g_j(\vec{x}^{\,q-1}) + \left(\frac{dg_j(\vec{x}^{\,q-1})}{d\alpha^*} \right) \bullet \alpha^* = 0.0$$

Equation D-44.

and an estimate for α^* is

$$\alpha^*_{est} = -\frac{g_j(\vec{x}^{\,q-1})}{\left(\frac{dg_j(\vec{x}^{\,q-1})}{d\alpha^*} \right)}$$

Equation D-45.

Therefore, even at the beginning of the one-dimensional search, a considerable amount of information is available to direct the process. Using the estimates for α^* given by Eq. D-43 and Eq. D-45and for each noncritical constraint, the smallest positive proposed α^* is taken as the first estimate of how far to move.

If constraints are currently violated, Eq. D-45 can still be used. Applying this approximation to all violated constraints, the largest proposed value of α^* is selected as an estimate of how far to move to overcome all constraint violations. If the projected move to a constraint that is not currently active is smaller than this, then a move to that constraint is made as a first estimate, since we do not want to add new constraints to the set of violated constraints.

Using similar approximations, we can also estimate an upper bound on α^* that forces all design variables to their lower or upper bounds. This then provides a maximum allowable value for α^*.

While this brief discussion is not detailed enough to cover all of the intricacies of estimating the first step in the one-dimensional search, it does provide a sense of the complexity of the decision processes. Also, it should be noted that on subsequent search directions, this process is modified to account for knowledge gained on previous iterations. For example, if an average of 5% improvement is obtained on the last two one-dimensional searches, then another 5% improvement is attempted on this one.

The one-dimensional search process now proceeds to find the bounds on the α^* that contain the solution. Once the bounds on α^* are known, the constrained minimum is found by interpolation. Since $\vec{S}^{q}$ has been defined as a direction of improving design, the search can be limited to positive values of α^*.

D.4 Finding Bounds On α^*

At the beginning of this process, the values of the objective and constraints are known at $\alpha^* = 0$ and $\alpha^* = \alpha_1$, where α_1 is the initial estimate for α^*. Following are three cases that must be considered when deciding if α_1 is the upper bound α_u.

Case 1 - All constraints are initially satisfied (all $g_j(\alpha^* = 0) <$ CTMIN). In this case, the search direction is the one that reduces the objective. Therefore, if at α_1, the objective is greater than at $\alpha_0(\alpha^* = 0)$, then α_1 is an upper bound on α^*. Also, since the initial constraints are all satisfied, if any $g_j(\alpha_1) >$ CTMIN, then α_1 is an upper bound, because the search has moved into the infeasible design space. If either the objective function has begun to increase or some constraint has become violated, then α_1 is chosen as the upper bound α_u.

Case 2 - One or more constraints are violated (greater than CTMIN) at $\alpha^* = 0$, and the objective function is increasing ($dF(x^{q-1})/d\alpha^* > 0$). Here, a search direction has been chosen that should reduce the constraint violation. Thus, at α_1, if the maximum constraint value is greater than the maximum constraint value at $\alpha^* = 0$, then α_1 is the upper bound α_u. This usually occurs when some new constraint becomes violated at α_1. If this happens, it is probably not possible to overcome the constraint violations in this direction, but the maximum constraint violation may at least be reduced. Alternatively, a value of α_1 might be found where all constraints are satisfied. If this happens, a feasible design is found and so, again, α_1 is an upper bound on α^*. Here, there is little concern that the objective function is increasing, since the first priority is to overcome the constraint violations. If the constraint violations can be overcome, it is desirable to do so with a minimum increase in the objective function.

Case 3 - One or more constraints are violated (greater than CTMIN) at $\alpha^* = 0$, and the objective function is decreasing ($dF(x^{q-1})/d\alpha^* > 0$). This is the same as Case 2 except that the objective function can be reduced. Therefore, everything is the same except that the search is not stopped if the constraint violation(s) are overcome. If a feasible design can be found that allows further reduction of the objective function, movement continues in this direction until some new constraint becomes active or the objective begins to increase. The corresponding α^* is the upper bound α_u.

During the process of finding the upper bound α_u on α^*, it may be required to move further than the initial step α_1. If α_1 does not yield an upper bound, another step $\alpha_2 = 2^*\alpha_1$ is taken and tried again. In practice, it may take many steps before a bound is found on the solution. If this is the case, the best four solutions are retained and called α_L, α_1, α_2, and α_U. If more than three steps are required, the lower bound α_L will not equal zero. Instead, a lower bound is carried forward to retain the nearest four proposed values of α^*. If fewer than four values are available (including $\alpha^* = 0$), then all are retained. For each retained value of α^*, the value of the objective and all constraint functions are also retained. These are then used to interpolate for an improved solution to the value of α^*.

D.5 Interpolation for α^*

Once bounds on the solution to the one-dimensional search problem have been established, it is desirable to refine the solution as much as possible. To achieve this, a polynomial interpolation of the objective and constraint functions is used. The basic tool used here is a simple polynomial curve fit, which may be linear, quadratic, or cubic, depending on the amount of information available. If more than four pieces of information are available, then higher order approximations could be used. However, experience has shown that a cubic fit is adequate to approximate the functions without introducing too much numerical error.

Whether the objective function or a constraint is to be approximated, the basic approach is the same. The process is demonstrated using a quadratic fit to three function values. The other approximations are similar and are explained in detail in Vanderplaats, “An Efficient Feasible Direction Algorithm for Design Synthesis,”[3].

A quadratic polynomial is defined as

$$F = a_0 + a_1 x + a_2 x^2$$

Equation D-46.

where: a_0, a_1, a_2 are constants, and x is the variable.

In this case, x corresponds to the one-dimensional search parameter α^*.

Next, assume that three values of F exist for three values of x, as listed in the table below:

x	F
0.5	–1.25
1.0	–3.00
2.0	–2.00

Substituting the values of F and x into Eq. D-46 results in three equations with three unknowns

$$a_0 + 0.5 \bullet a_1 + 0.25 \bullet a_2 = -1.25$$

Equation D-47.

$$a_0 + 1.0 \bullet a_1 + 1.00 \bullet a_2 = -3.0$$

Equation D-48.

$$a_0 + 2.0 \bullet a_1 + 4.00 \bullet a_2 = -2.0$$

Equation D-49.

Solving for the values of a_0, a_1, and a_2 results in

$$a_0 = 2.0,\ a_1 = -8.0,\ a_2 = 3.0$$

Equation D-50.

Substituting this into Eq. D-46 results in

$$F = 2.0 - 8.0x + 3.0x^2$$

Equation D-51.

Eq. D-51 can be used in different ways depending on whether the function F is the objective or a constraint. First, assume that this is the objective function and we wish to find the value of x that minimizes it. The function F has a minimum or maximum value when the derivative of F with respect to x is zero. Differentiating Eq. D-51 with respect to x yields:

$$\frac{dF}{dx} = -8.0 + 6.0 \bullet x$$

Equation D-52.

Setting this to zero and solving for *x* gives a proposed solution *x* = 1.33. Substituting this into Eq. D-51 gives an estimated value of the function of *F* = –3.33. However, this is not sufficient to indicate that *F* is a minimum. The second derivative must also be considered, where

$$\frac{d^2F}{dx^2} = 6.0$$

Equation D-53.

Since the result is positive, the proposed solution represents a minimum of *F* instead of a maximum. You can verify the results by a rough plot of the original three points given and passing them over a quadratic curve.

Now, assume that the function to be approximated is a constraint. In this case, the value of *x* is sought that drives the function to zero (i.e., forces the design to a constraint boundary). To achieve this, the value of *x* must be found that makes Eq. D-51 equal to zero. Setting Eq. D-51 to zero and solving for *x* yields

$$x = \frac{[4.0 + \sqrt{10.0}]}{3.0} = 0.2792,\ 2.3874$$

Equation D-54.

For each of these values of *x*, the estimated value of *F* is zero. The question is: Which one should be chosen? In this case, note that the initial value *F*(*x* = 0) is positive. Therefore,*x* = 0.2792 is a proposed lower bound on α^*, since this overcomes the constraint violation (to a quadratic approximation). Thus, this is a possible lower bound on α^*, α_L. However, more information is available. The value of the variable *x* = 2.3874 is the point beyond which the constraint again becomes violated. Therefore, this value of *x* is a possible upper bound on α^*, α_U. This can be verified by calculating the gradient of *F* at α_L and α_U to determine if it is negative at α_L, (i.e., forcing g_j to be more negative) and positive at α_U (i.e., forcing g_j to be violated). To verify this, simply substitute the values of *x* into Eq. D-52, or examine the plot of the function.

The optimization process leads to an estimate for α^* for the minimum of the objective and the zero of one constraint. In this case, the constraint is initially violated so we have both a lower and an upper bound on α^*. However, we must also estimate values of α^* for all constraints. The maximum lower bound and the minimum upper bound define the range for the proposed solution to the one-dimensional search. It may happen that the proposed upper bound is less than the proposed lower bound. If this happens, it usually means that for each proposed α^* there are one or more violated constraints. In this case, Eq. D-52 is used to minimize the maximum constraint violation, and this decision overrides other conditions.

Let us assume that the polynomial approximation for the objective and all constraints is used and concludes with a lower and upper bound on the constraint values so that the upper bound is greater than the lower bound. (If no constraints are violated at $\alpha^* = 0$, this is the usual case.) Next a choice must be made as to which value of to select. If the search starts with violated constraints and the lower bound is less than the upper bound, either the lower bound is selected, or the value is chosen that minimizes the objective (assuming that it is between the lower and upper bounds). If no constraints are violated at $\alpha^* = 0$, the minimum proposed α^* is selected from the α^* that minimizes the objective and the α^* that represents the upper bound.

As indicated by the difficulty in describing the one-dimensional search process, this is a particularly complex part of the optimization process. The description given here is only a simple outline of the process. In practice, decisions must be made based on how many proposed values of α^* are available using linear, quadratic and cubic curve fits to estimate the best value of the move parameter. Since it is impossible to predict all the possible combinations of the situations that may arise, this is necessarily a search process that is based more on experience than theory. The key is the need to consistently improve the design. If the absolute best solution does not appear during this one-dimensional search, further iterations provide other chances until the convergence criteria are satisfied. This explains why more complex methods are not used (with their associated number of function evaluations and cost) during the one-dimensional search. In the optimization algorithm used in NX Nastran, the one-dimensional search process is somewhat more sophisticated but basically follows the same process as outlined here.

At this point, the best search direction to improve the design has been determined and a search conducted in that direction. The question now is: How can it be determined if the optimal design has been achieved? The answer is based on the decisions that determine when to terminate the optimization process.

D.6 Convergence to the Optimum

Since numerical optimization is an iterative process, one of the most critical and difficult tasks is determining when to stop. The optimization software uses several criteria to decide when to end the iterative search process, and these are described here. It is important to remember that the process described in this section relates only to the solution of the approximate optimization problem. The number of cycles through the entire design process is controlled by other similar criteria.

Maximum Iterations

As with any iterative process, a maximum iteration counter is included. The default for this is 40 iterations (search directions). Usually, an optimum is found sooner than this; therefore, the maximum is mainly intended to avoid excessive computations.

No Feasible Solution

If the initial design is infeasible (constraints are violated), the first priority is to overcome these violations and find a feasible solution. However, if there are conflicting constraints, a feasible solution may not exist. Therefore, if a feasible design is not achieved in 20 iterations, the optimization process is terminated.

Point of Diminishing Returns

Probably the most common situation is where the optimum is approached asymptotically. Therefore, while some progress is still being made, continued iterations are not justified. This is particularly true in NX Nastran because this program is solving an approximate problem that is to be updated on the next design cycle. Here, two criteria are used. The first criterion requires that the relative change in the objective between iterations be less than a specified tolerance DELOBJ. Thus, the criterion is satisfied if:

$$\frac{\left|F(\vec{x}^{\,q}) - F(\vec{x}^{\,q-1})\right|}{\left|F(\vec{x}^{\,q-1})\right|} \le \text{DELOBJ}$$

Equation D-55.

The default value for DELOBJ is 0.001.

The second criterion is that the absolute change in the objective between the iterations is less than a specified tolerance DABOBJ. This criterion is satisfied if

$$\left|F(\vec{x}^{\,q}) - F(\vec{x}^{\,q-1})\right| \le \text{DABOBJ}$$

Equation D-56.

The default value for DABOBJ is the maximum of $0.001 \bullet \left|F(\vec{x}^{\,0})\right|$ and 1.0E-20.

The reason for the two criteria is that if the objective function is large, the relative change between two successive iterations is an indication of convergence. However, if *F*(*x*) is a very small number, a relative change is not meaningful, and thus the absolute change controls the convergence.

Often, no progress is made on one iteration, but significant progress is produced on the next. Therefore, one of these criteria is required to be satisfied on ITRMOP consecutive iterations where the default value of ITRMOP is 2. Recall that this default may be changed on the DOPTPRM Bulk Data entry.

Satisfaction of the Kuhn-Tucker Conditions

The Kuhn-Tucker conditions necessary for optimality are stated in Eq. D-14 through Eq. D-17. In the uncommon case where there are no constraints, these conditions degenerate to the familiar case of the vanishing gradient of the objective. In practice, this is assumed to be true if all components of the gradient are less than 0.001.

The usual optimization task in NX Nastran has many constraints imposed. In the process of finding a usable-feasible search direction, it is possible to detect if the Kuhn-Tucker conditions are satisfied. If they are, the optimization process should be terminated. However, in practice, other considerations may affect the results. Remember that a constraint is defined as active if it is numerically greater than the parameter CT. The optimization starts with CT = 0.03. (Any constraint within three percent of being satisfied is called critical.) Since a constraint with this value is considered in the Kuhn-Tucker calculations, these conditions may be satisfied even if the search is up to three percent away from the constraint boundary. Therefore, if CT is greater than CTMIN in magnitude, the CT value is reduced by some fraction (typically 50%) and another attempt is made to find a usable-feasible direction. If CT is reduced in magnitude to a value of –CTMIN, and can still satisfy the Kuhn-Tucker conditions, the optimization process is terminated.

D.7 Sequential Linear Programming

Sequential Linear Programming (see reference 10) was briefly introduced in Structural Optimization. This section presents the method in greater detail.

The basic concept is actually quite simple. First, Taylor Series approximations are created for the objective and constraint functions. These approximations are then used for optimization, instead of the original nonlinear functions. Now when the optimizer requires the values of the objective and constraint functions, these are very easily and inexpensively calculated from the linear approximation. Also, since the approximate problem is linear, the gradients of the objective and constraints are available directly from the Taylor Series expansion.

Traditionally, the SIMPLEX algorithm[10] is used for solving the linear approximate problem. However, the optimizer in NX Nastran uses the Modified Method of Feasible Directions, since it solves linear problems nicely and works well for the size of problems normally considered.

The general algorithm proceeds in the following steps:

1. For the current values of the design variables, sort the constraints and retain the most critical for use during this cycle. Typically, about 5 • NDV constraints are retained. The reason that all constraints are not retained is that there may be thousands, and most may be far from critical. Therefore, it would be wasted effort to calculate their gradients.

2. Create first order Taylor Series expansions of the objective and retained constraints with respect to the design variables.

3. Define move limits on the design variables. Typically, during one cycle, the design variables will be allowed to change by 20-40%, but this is adjusted during later cycles.

4. Solve the linear approximate optimization problem.

5. Check for convergence. If satisfied, EXIT. Otherwise, repeat the process from step 1.

In step 2, Taylor Series expansions are created in the form:

$$\tilde{F}(\vec{x}) = F(\vec{x}^{\,q-1}) + \nabla F(\vec{x}^{\,q-1}) \cdot \delta\vec{x}$$

Equation D-57.

$$\tilde{g}_j(\vec{x}) = g_j(\vec{x}^{\,q-1}) + \nabla g_j(\vec{x}^{\,q-1}) \cdot \delta\vec{x} \qquad j \in J$$

Equation D-58.

10. Dantzig, G. B., *Linear Programming and Extensions*, Princeton University Press, Princeton, NJ, 1963.

$$\text{where } \delta\vec{x} = \vec{x}^{q} - \vec{x}^{q-1}$$

Equation D-59.

and *J* is the set of retained constraints.

Note that everything in Eq. D-57 and Eq. D-58 is constant except the values of the design variables, $\vec{x}^{q}$. Therefore, Eq. D-57 and Eq. D-58 can be rewritten as

$$\tilde{F}(\vec{x}^{q}) = \tilde{F}^{0} + \nabla F(\vec{x}^{q-1}) \cdot \vec{x}^{q}$$

Equation D-60.

$$\tilde{g}_j(\vec{x}^{q}) = \tilde{g}_j^{0} + \nabla g_j(\vec{x}^{q-1}) \cdot \vec{x}^{q} \qquad j \in J$$

Equation D-61.

$$\text{where } \tilde{F}^{0} = F(\vec{x}^{q-1}) - \nabla F(\vec{x}^{q-1}) \cdot \vec{x}^{q-1}$$

Equation D-62.

$$\text{and } \tilde{g}_1^{0} = g_j(\vec{x}^{q-1}) - \nabla g_j(\vec{x}^{q-1}) \cdot \vec{x}^{q-1}$$

Equation D-63.

The following linear approximate optimization problem is now solved:

$$\text{minimize} \qquad \tilde{F}(\vec{x}^{q})$$

Equation D-64.

$$\text{subject to} \quad \tilde{g}_j(\vec{x}^{q}) \le 0 \qquad j \in J$$

Equation D-65.

$$\tilde{x}_i^{L} \le x_i^{q} \le \tilde{x}_i^{U} \qquad i = 1, N$$

Equation D-66.

$$\text{where } \tilde{x}_i^L = x_i^q - D\left|X_i^q\right|$$

Equation D-67.

$$\text{and } \tilde{x}_i^U = x_i^q + D\left|X_i^q\right|$$

Equation D-68.

The multiplier *D* in Eq. D-67 and Eq. D-68 is initially set to RMVLMZ. Depending on the progress of the optimization, this parameter will be sequentially reduced. Typically, the first approximate optimization produces a design that violates one or several constraints. However, as the optimization proceeds, this constraint violation should be reduced. If it actually increases, then *D* is reduced, as well as the move limits on the design variables.

The convergence criteria for the SLP method are much the same as for the other methods, with the addition that the process will not terminate if there are significant constraint violations. In this case, the process continues until the constraints are satisfied or the maximum number of iterations is reached.

D.8 Sequential Quadratic Programming

This method is considered to be an excellent method by many theoreticians[11] [12].

The basic concept is very similar to Sequential Linear Programming. First the objective and constraint functions are approximated using Taylor Series Approximations. However, a quadratic, rather than a linear approximation of the objective function is used. Linearized constraints are used with this to create a direction finding problem of the form:

$$\text{minimize } Q(S) = F^0 + \nabla F \bullet S + 0.5 S^T B S$$

Equation D-69.

$$\text{subject to } \nabla g_j \bullet S + g_j^0 \le 0 \qquad j = 1, m$$

Equation D-70.

This sub-problem is solved using the Modified Method of Feasible Directions. The matrix *B* is a positive definite matrix that is initially the identity matrix. On subsequent iterations, *B* is updated to approach the Hessian of the Lagrangian functions.

Now assume the approximate problem of minimizing *Q* subject to the linearized constraints has been solved. At the optimum for this problem, the Lagrange multipliers λ_j (j = 1, m) can be calculated. With

11. Powell, M. J. D., “Algorithms for Nonlinear Constraints that use Lagrangian Functions,” Math. Prog., Vol. 14, No. 2, pp. 224–248, 1978.,
12. Vanderplaats, G. N. and Sugimoto, H., “Application of Variable Metric Methods to Structural Synthesis,” Engineering Computations, Vol. 2, No. 2, June, 1985.

the Lagrange multipliers available, an approximate Lagrangian function can be constructed and used to search in the direction defined by $\vec{S}$. The task is to find α in order to:

$$\text{minimize } \Phi = F(\vec{x}) + \sum_{j=1}^{m} u_j \max[0, g_j(\vec{x})]$$

Equation D-71.

where:

$\vec{x}$	=	$\vec{x}^{q-1} + \alpha\vec{S}$	
u_j	=	$\lvert\lambda_j\rvert$	j = 1, m first iteration
u_j	=	$\max\left[\lvert\lambda_j\rvert, \frac{1}{2}(u_j' + \lvert\lambda_j\rvert)\right]$	j = 1, m subsequent iterations

and $u_j' = u_j$ from the previous iteration.

During the one-dimensional search, approximations are made to the components of Φ because this function has discontinuous derivatives at the constraint boundaries, but the components are smooth.

After the one-dimensional search is complete, the B matrix is updated using the BFGS formula;

$$[B^*] = [B] - \frac{[B]\{p\}\{p\}^T[B]}{\{p\}^T[B]\{p\}} + \frac{\{\eta\}\{\eta\}^T}{\{p\}^T\{\eta\}}$$

Equation D-72.

where:

$$\{p\} = \vec{x}^q - \vec{x}^{q-1}$$

$$\{\eta\} = \theta\{y\} + (1 - \theta)[B]\{p\}$$

$$\{y\} = \nabla x\,\Phi^q - \nabla x\,\Phi^{q-1}$$

$$\Phi = F(\vec{x}) + \sum_{j=1}^{m} \lambda_j g_j(\vec{x})$$

$$\theta = \begin{cases} 1.0 & \text{if } \{p\}^T\{y\} \geq 0.2\{p\}^T[B]\{p\} \\ \dfrac{0.8\{p\}^T[B]\{p\}}{\{p\}^T[B]\{p\} - \{p\}^T\{y\}} & \text{if } \{p\}^T\{y\} < (0.2\{p\}^T[B]\{p\}) \end{cases}$$

Now [B^*] replaces [B] and a new iteration is begun.

D.9 Summary

The purpose of this Appendix has been to provide an overview of the computational details of the optimization process. The techniques used in the optimizer have been chosen because they are reasonably robust and because they allow for considerable flexibility in formulating the design problem.

From the variety of decisions that must be made during the optimization process it is clear that anything the user can do to improve the numerical conditioning of the problem will have a strong influence on the efficiency and reliability of the result. It is hoped that some understanding of the actual optimization process will assist in this.

Siemens Industry Software

Headquarters
Granite Park One
5800 Granite Parkway
Suite 600
Plano, TX 75024
USA
+1 972 987 3000

Americas
Granite Park One
5800 Granite Parkway
Suite 600
Plano, TX 75024
USA
+1 314 264 8499

Europe
Stephenson House
Sir William Siemens Square
Frimley, Camberley
Surrey, GU16 8QD
+44 (0) 1276 413200

Asia-Pacific
Suites 4301-4302, 43/F
AIA Kowloon Tower, Landmark East
100 How Ming Street
Kwun Tong, Kowloon
Hong Kong
+852 2230 3308

About Siemens PLM Software

Siemens PLM Software, a business unit of the Siemens Industry Automation Division, is a leading global provider of product lifecycle management (PLM) software and services with 7 million licensed seats and 71,000 customers worldwide. Headquartered in Plano, Texas, Siemens PLM Software works collaboratively with companies to deliver open solutions that help them turn more ideas into successful products. For more information on Siemens PLM Software products and services, visit www.siemens.com/plm.