

MSC Nastran 2017

Design Sensitivity and Optimization User's Guide

11/25/16



Main Index

Corporate

MSC Software Corporation
4675 MacArthur Court, Suite 900
Newport Beach, CA 92660
Telephone: (714) 540-8900
Toll Free Number: 1 855 672 7638
[Email:](mailto:americas.contact@mscsoftware.com) americas.contact@mscsoftware.com

Japan

MSC Software Japan Ltd.
Shinjuku First West 8F
23-7 Nishi Shinjuku
1-Chome, Shinjuku-Ku
Tokyo 160-0023, JAPAN
Telephone: (81) (3)-6911-1200
[Email:](mailto:MSCJ.Market@mscsoftware.com) MSCJ.Market@mscsoftware.com

Worldwide Web

www.mscsoftware.com

Disclaimer

MSC Software Corporation reserves the right to make changes in specifications and other information contained in this document without prior notice.

The concepts, methods, and examples presented in this text are for illustrative and educational purposes only, and are not intended to be exhaustive or to apply to any particular engineering problem or design. MSC Software Corporation assumes no liability or responsibility to any person or company for direct or indirect damages resulting from the use of any information contained herein.

User Documentation: Copyright © 2016 MSC Software Corporation. Printed in U.S.A. All Rights Reserved.

This notice shall be marked on any reproduction of this documentation, in whole or in part. Any reproduction or distribution of this document, in whole or in part, without the prior written consent of MSC Software Corporation is prohibited.

This software may contain certain third-party software that is protected by copyright and licensed from MSC Software suppliers. Additional terms and conditions and/or notices may apply for certain third party software. Such additional third party software terms and conditions and/or notices may be set forth in documentation and/or at <http://www.mscsoftware.com/thirdpartysoftware> (or successor website designated by MSC from time to time).

PCGLSS 8.0, Copyright © 1992-2016, Computational Applications and System Integration Inc. All rights reserved. PCGLSS 8.0 is licensed from Computational Applications and System Integration Inc.

MSC, Dytran, Marc, MSC Nastran, Patran, the MSC Software corporate logo, e-Xstream, Digimat, and Simulating Reality are trademarks or registered trademarks of the MSC Software Corporation and/or its subsidiaries in the United States and/or other countries.

NASTRAN is a registered trademark of NASA. LS-DYNA is a trademark or registered trademark of Livermore Software Technology Corporation. FLEXIm and FlexNet Publisher are trademarks or registered trademarks of Flexera Software. All other trademarks are the property of their respective owners.

Use, duplicate, or disclosure by the U.S. Government is subjected to restrictions as set forth in FAR 12.212 (Commercial Computer Software) and DFARS 227.7202 (Commercial Computer Software and Commercial Computer Software Documentation), as applicable.

U.S. Patent 9,361,413

Revision 0. November 25, 2016

NA:V2017:Z:Z:DC-DSO-PDF

Europe, Middle East, Africa

MSC Software GmbH
Am Moosfeld 13
81829 Munich, Germany
Telephone: (49) 89 431 98 70
[Email:](mailto:europe@mscsoftware.com) europe@mscsoftware.com

Asia-Pacific

MSC Software (S) Pte. Ltd.
100 Beach Road
#16-05 Shaw Tower
Singapore 189702
Telephone: 65-6272-0082
[Email:](mailto:APAC.Contact@mscsoftware.com) APAC.Contact@mscsoftware.com



Contents

Design Sensitivity and Optimization User's Guide

Preface

About this Book	ii
Organization	iii
Developments Introduced Since Version 68 of MSC Nastran	iv
List of MSC Nastran Books	viii
Technical Support	ix
Training and Internet Resources	x

1 Getting Started

Introduction	2
What is Design Optimization?	2
What is Design Sensitivity?	2
Why Use Design Sensitivity and Optimization?	3
How Does Design Optimization Differ from Analysis?	4
Numerical Optimization Basics	9
How Much Do I Need to Know About Optimizers?	9
An Illustration	9
Numerically Searching for an Optimum	13
Structural Optimization	22
Relating the Finite Element Analysis Model to the Design Model	22
Managing the Structural Optimization Task	22
Overview of Approximation Concepts Used in Structural Optimization	23
Summary	26

2 Fundamentals of Design Sensitivity and Optimization in MSC Nastran

Overview of Fundamentals	28
Initial Design	29



Structural Analysis	29
Constraint Screening	30
Sensitivity Analysis	30
Optimizer	30
Approximate Model	30
The Improved Design	30
Converged	31
Types of Optimization	32
Sizing Optimization	32
Shape Optimization	32
Toptimization	32
Topology Optimization	32
Topometry Optimization	32
Topography Optimization	32
The Design Model	33
Design Variables	33
Designed Properties	33
Type-1 Properties	36
Type-2 Properties	38
Type-3 Properties	38
Designed Shapes	39
Design Responses	39
Type-1 Responses	39
Type-2 Responses	39
Type-3 Responses	40
Multidisciplinary Analysis	43
Response Calculation	44
Mode Tracking	46
Constraint Screening	47
Design Sensitivity Analysis	50
General Considerations	50
Element and Grid Responses	53
Optimization	74
Connection Between the Optimizer and the Approximate Model	74
The Approximate Model	78
Formal Approximations	78
A Simple Linear Design Space	79
Move Limits	81
Design Variable Limits	81



Property Limits	82
Transformation of the Approximate Optimization Task to a Feasible Design	
85	
Function Evaluation	86
Gradient Evaluation	93
Tests for Convergence	98
Convergence of Design Cycles: Hard and Soft Convergence	98

3 Developing the Design Model for Sensitivity and Optimization

Overview of Design Modeling	109
Case Control for Design Optimization	111
Defining the Design Variables	113
Relating Design Variables to Properties	116
Designating the Design Responses	119
DRESP1 Bulk Data Entries	119
DRESP2 Bulk Data Entries	119
DRESP3 Bulk Data Entries	125
Defining the Objective Function	127
Defining the Constraints	129

4 Input Data

Optimizers (Licensing)	133
File Management	134
Shape Optimization	134
User Defined Beam Libraries	134
External Responses	135
Executive Control	136
Solution 200	136
Case Control Section	137
Analysis Discipline Definition	137
Design Task Definition	138
Mode Tracking	140
Shape Basis Vector Computation	142
Generation of a New Bulk Data File	142



Bulk Data Entries	143
BEADVAR	143
BNDGRID	146
DCONADD	147
DCONSTR	148
DDVAL	151
DEQATN	152
DESVAR	155
DLINK	157
DOPTPRM	159
DRESP1	159
DRESP2	165
DRESP3	170
DSCREEN	174
DTABLE	176
DTABLE2	177
DVBSHAP	178
DVCREL1	179
DVCREL2	182
DVGRID	184
DVLREL1	185
DVMREL1	187
DVMREL2	190
DVPREL1	192
DVPREL2	194
DVPSURF	197
DVSHAP	197
MODTRAK	198
SEDLINK	199
SEDRSP3	203
STOCHAS	206
TOMVAR	207
TOPVAR	209
Parameters Unique to Design Sensitivity and Optimization	212
Design Responses	215
DRESP1	215
Optimization Parameters	235
DOPTPRM	235
External Response	245
External Response or Type-3 Response	245



User Interface for Setting Up External Response	248
Creating a DRESP3 entry	248
Write C or Fortran Routines to Calculate Response Based on Template Routines	249
Build External Servers using SCons Tool	254
Creating a CONNECT Command	260
Creating an Evaluator Connection File	261
Submitting a Nastran Job with the gmconn Keyword	261

5 Output Features and Interpretation

Output-Controlling Parameters	267
Design Optimization Output	269
DSOUG4 Output	269
Modification of Move Limit Parameters	286
Special Prints for Fully Stressed Design	287
Special Prints for Discrete Variable Optimization	288
Special Prints for Topology Optimization	290
Design Sensitivity Output	291
Formatted Design Sensitivity	291
Unformatted Design Sensitivity	294
Design Punch Output	297
Punch Parameters	297
Example for DESPCH	298
Special Punch Considerations for Topology Optimization	299
Special Punch in the Case of Shape Optimization	299
New Bulk Data File	300
Postprocessing Output	303
Comma Separated Values File	303
Output from PARAM POST	304

6 Shape Optimization

Introduction	310
Basis Vectors in Shape Optimization	311
Relating Design Variables to Shape Changes	316
Use of Patran to generate a Shape Optimization deck	336
Examples	343



Shape Optimization of a Culvert	343
Analytic Boundary Shapes	353

7 Topology, Topometry and Topography Optimization

Topology Optimization	372
Introduction	372
Features and Benefits	372
Brief Literature Review	373
Bulk Data and Parameters	375
TOPVAR	375
Bulk Data Entry DRESP1	378
Design Optimization Parameters (DOPTPRM)	379
Modeling Guidelines and Limitations	384
Modeling Tips	384
Limitations	386
Patran User Interface	387
General Topology Optimization Preprocessing	387
Patran Postprocessing	391
Application Examples	395
Bridge Example	395
MBB Beam with Variations	414
Torsion Beam with Variations	419
Engine Mount	425
Optimization Solution	427
Topology Optimization with Glued Contact	450
Topometry Optimization	457
Introduction	457
Benefits	457
TOMVAR	458
Output	460
Guidelines and Limitations	460
Example 1 - Three-bar Truss (tomex1.dat)	460
Input	462
Output	463
Example 2 – Car Model Topometry Design	464
Post-processing Topometry Results	464
Topography (Bead or Stamp) Optimization	468



Introduction	468
Benefits	468
BEADVAR	468
Outputs	472
Guidelines and Limitations	472
Example 3 – A Square (togex1.dat)	473
Input	473
Output	474
Postprocessing Topography Results	475
Toptimization	478
Example 1 - Topography	478
Example 2 - Topometry Optimization	483

8 Example Problems

Introduction	488
Three-Bar Truss	490
Analysis Model Description	491
Design Model Description	491
Vibration of a Cantilevered Beam (Turner's Problem)	513
Analysis Model Description	514
Design Model Description	514
Cantilevered Plate	518
Analysis Model Description	519
Design Model Description	519
Stiffened Plate	527
Analysis Model Description	528
Design Model Description	531
Dynamic Response Optimization	533
Twenty-Five Bar Truss, Superelement and Discrete Variable Optimization	544
Analysis Model Description	544
Design Model Description	545
Design Optimization with Composite Materials with Fully Stressed Design	552
Acoustic Optimization	559



RMS Response	567
Transient Dynamic Optimization	574
External Response to Include Alternative Buckling Response	579
Continuing the Design Process in a Subsequent Job	590
Continuing a Property Optimization Design Task	590
Continuing a Shape Optimization Design Task	593
Restarting from a Previous Analysis	594
Cantilevered Plate with Redundant Materials	598
Analysis Model Description	598
Design Model Description	599

9 Special Topics

Introduction	606
Discrete Variable Optimization	606
Fully Stressed Design	608
Trust Region	611
Superelement Optimization	617
Automatic External Superelement Optimization (AESO)	622
Randomization of a User's Input Data File	631
Optimization of Nonlinear Structural Responses (Pre-release)	633
Solving Large Problems in SOL 200	655
Special Considerations When Designing One-Dimensional Bending Elements	660
MultiOpt-Global Optimization and Multi Model Optimization	664
Introduction	664
Benefits	665
User Input	666
User Output	671
GO Test Case	671
MMO Test Case	674
Guidelines and Limitations	676
OpenMDO™	679



A Nomenclature, Glossary of Terms, and References

Nomenclature	682
Glossary of Terms	685
References	693

B Adding Your Own Beam Cross-Section Library

Introduction	696
BSCON Subroutine	698
BSBRPD Subroutine	700
BSGRQ Subroutine	711
BSBRT Subroutine	712
BSBRID Subroutine	715
BSBRGD Subroutine	721
BSBRCD Subroutine	727
BSMSG Subroutine	733
Linking Your Library to MSC Nastran	734
Example of Building and Linking a Beam Server	735

C The IPOPT Algorithm

IPOPT	738
Introduction	738
Benefits	738
Theory	738





Preface

- About this Book ii
- List of MSC Nastran Books viii
- Technical Support ix
- Training and Internet Resources x



About this Book

MSC Nastran is a general purpose finite element program which solves a wide variety of engineering problems. It is developed, marketed and supported by the MSC Software Corporation.

The *MSC Nastran 2014.1 Design Sensitivity and Optimization User's Guide* is intended to explain the capability of MSC Nastran to predict the effect of changes in the structural model on structural responses (sensitivity) and to resize the structure so as to satisfy imposed design conditions while taking a particular response to minimum (or maximum) value (Optimization). Note that sensitivity analysis is available in standard MSC Nastran and does not require additional licenses. The resizing capability does require Nastran Optimization license. Guidance is provided in preparing the input and assessing the output. A number of examples are provided to illustrate the use of MSC Nastran Design Sensitivity and Optimization.

This guide has been made possible by the work of numerous people at the MSC Software Corporation. Notably, the Optimization Project within the Nastran Development organization provided the technical developments of the capability. Past and present members of this group are Dr. Erwin Johnson, Dr. Gee (David) Chou, Dr. Shenghua Zhang, Dr. Xiaoming Yu and Dr. Swaminathan Natarajan. Mr. Mike Reymond has led the development effort associated with the DMAP (Direct Matrix Abstraction Program) used in the implementation of this capability. Dr. Srinivas Kodiyalam provided MSC Software with his enhanced version of the ADS source code at no cost.

This guide contains many highlighted links (text in blue) to other MSC Nastran documents. All the documents are delivered together as a collection. The links between the documents will work, if you keep the collection together.

Two ways of working with links are as follows:

- Use “alt ← “ to return back to the window your cursor is in.
- To open the other “linked to” documents in a new window from an Adobe Reader, choose Edit → Preferences → Documents → Open cross-document links in the same window. Then uncheck the checkbox and select **OK**.



Organization

The Guide is divided into nine chapters and three appendices. A brief description of each of these is:

Chapter 1. Getting Started

This chapter provides an overview of design sensitivity and optimization and is presented at an introductory level. This chapter assumes limited knowledge of the field and can be skimmed or skipped altogether by knowledgeable workers in this area.

Chapter 2. Fundamentals of Design Sensitivity and Optimization in MSC Nastran

This chapter can be regarded as a theoretical overview of the Design Sensitivity and Optimization capability that identifies and describes the concepts that are used. There is minimal reference to the user interface in this chapter. Instead, it provides information on the algorithms used and is intended for the high-end user and for developers.

Chapter 3. Developing the Design Model for Sensitivity and Optimization

This chapter serves as a bridge between the concepts of Chapter 2 and the input description of Chapter 4. It is intended to provide an overview of the concept of the Design Model as it is implemented in MSC Nastran and provides guidelines for the use of the capabilities.

Chapter 4. Input Data

This chapter provides explicit descriptions of each of the Case Control Commands, Bulk Data Entries and Parameters that are related to design sensitivity and optimization.

Chapter 5. Output Features and Interpretation

This chapter provides a description of the output available from running design sensitivity and optimization tasks. In addition to a comprehensive discussion of the output provided in the .f06 output file, there is also mention of the design sensitivity and optimization outputs that are available in the .pch file and that can be used in post-processors, especially Patran.

Chapter 6. Shape Optimization

This chapter focuses exclusively on shape optimization. The reader can use this as a standalone reference for shape optimization. A discussion of Basis Vectors, various methods available to generate basis vectors, various sections of the input deck unique to shape optimization, example problems and Patran support are the topics that make up this chapter.

Chapter 7. Topology, Topometry and Topography Optimization

This chapter is intended to provide a complete reference to topology, topometry and topography optimization. It contains the theory, MSC Nastran specific input deck entries, features and guidelines, a complete tutorial to Patran usage for pre- and post-processing and example problems.



Chapter 8. Example Problems

A series of example problems are given to illustrate the capabilities available in MSC Nastran for Design Sensitivity and Optimization. All the example problems are available in the MSC Nastran TPL (Test Problem Library) with the naming convention DSOUGxx.dat. It is expected that all readers will benefit from reading this chapter and practising the example problems. Studying the effects of variations on the provided input should be particularly beneficial.

Chapter 9. Special Topics

This final chapter is provided to go into greater depth on features that are part of MSC Nastran's design optimization capability, but that require special handling outside the descriptions of the more basic features.

Appendix A. Nomenclature and Glossary of Terms

This appendix provides a quick reference for the nomenclature used in this guide and to some technical terms used throughout. It also contains a listing of references called out in this guide.

Appendix B. Adding Your Own Beam Cross-section Library

Appendix C. The IPOPT Optimizer

This appendix provides a brief description of the theory contained in the IPOPT optimization algorithm.

Developments Introduced Since Version 68 of MSC Nastran

New versions of MSC Nastran are released periodically. This guide has been prepared with minimal reference to the version number and is consistent with the 2014.1 Release of MSC Nastran. It is felt to be of historical interest to show how the Design Sensitivity and Optimization capability evolved. It is also of great practical interest to those who are not using the most recent release to see which of the capabilities described in this Guide are not available to them. The major enhancements since Version 68 are identified here. Discussions of each of these topics are included in this Guide.

Version 68.2

- Support for inertial relief in a statics analysis
- Consideration of differential stiffness effects in buckling sensitivity analysis

Version 69

- Design of the PBARL. This was done in conjunction with the introduction of the beam library in the same release. The beam library also allowed for the analysis of PBEAML properties, but their design was deferred until the Version 70.7 release.
- Mode tracking
- Formatted design sensitivity prints



- Support for multiple boundary conditions (normal modes, buckling and flutter). Multiple static boundary conditions were first supported in Version 68.

Version 70

- Adjoint sensitivity analysis

Version 70.5

- Rigid element sensitivities with shape design changes
- Simplified design of beams

Version 70.7

- Design of the PBEAML
- Material and connectivity sensitivity
- Total weight response

In earlier releases, the weight was computed by multiplying the structural volume by the density of the material. This neglected any non-structural mass and concentrated masses.

- Extend the DRESP2

This entailed a number of new features. More significant ones include the ability to include material and connectivity properties and the ability to reference a DRESP2.

- Enhanced design data in the punch file
- Output of a comma separated values (CSV) file with design data that can be read into a spreadsheet

MSC Nastran 2001

- Complex eigenvalue sensitivity and optimization
- Discrete variable optimization
- Random response sensitivity and optimization
- Fully stressed design

MSC Nastran 2004

- External Response (DRESP3)
- Eigenvector sensitivity and optimization
- Subcase dependent frequency excitation sets

Prior to this enhancement, it was necessary to use the same set of excitation frequencies in each frequency response subcase.

- DMP and ACMS technology
- Punch of a complete, unsorted Bulk Data file
- Rayleigh Quotient Approximation for eigenvalues/eigenfrequencies



- Spanning of subcases and superelements
- PSD response sensitivity and optimization
- DESVAR Case Control command
- Removal of property limit constraints that are redundant with design variable limits

MSC Nastran 2005

- Topology optimization
- Composite strength ratio response
- Beta and match function for the DRESP2.
- ADS optimizer
- BIGDOT optimizer
- Transformation of approximate optimization task for a feasible design

MSC Nastran 2005 R2

- Topology Optimization
- Zero Density Material
- High's method for eigenvector sensitivity and optimization
- Design of PBRSECT and PBMSECT

MD Nastran R1 and MSC 2005 R3

- Topology optimization with manufacturability constraints
- Revised Optimizer options
- Supporting Trust Regions in SOL 200 for Adaptive Move Limits
- Support of analysis model value overriding design model value

MD Nastran R2 and MD Nastran R2.1

- Topology optimization enhancements
- Automatic External Superelement Optimization

MD Nastran R3 and MSC Nastran 2008 R1

- Enhancements in DRESP3
- Topometry Optimization
- Topography Optimization
- Permanent glued contact modeling in SOL 200
- Randomization of an Input data file
- Random elimination of element types
- Optimization of nonlinear structural responses (pre-release)



MD Nastran 2010

- Design of Part Superelements
- Integer input for DTABLE/DTABLE2
- Invariant DRESP3 gradients
- Design of monitor points
- Parallel design sensitivities
- IPOPT Optimization Algorithm
- Optimization of nonlinear response with topology and contact
- Multi-Model Optimization - MultiOPT
- BIGDOT and DOT are not supplied with MD Nastran 2010. Default Optimizers changed to MSCADS and IPOPT.

MSC NASTRAN 2011

- OpenMDO - allows user to insert their own Optimizer.

MSC NASTRAN 2012

- Design of Control Surfaces

MSC NASTRAN 2013

- ERP and Fatigue responses

MSC NASTRAN 2014

- Fatigue responses for Spot and Seam welds
- Design of Loads

MSC NASTRAN 2014.1

- Weight as a function of material or property ID

MSC NASTRAN 2016

- Global Optimization

MSC NASTRAN 2017

- Stress constraints in Topology Optimization
- Maximum thickness constraints in Topology Optimization



List of MSC Nastran Books

A list of some of the MSC Nastran documents is as follows:.

Installation and Release Guides

- Installation and Operations Guide
- Release Guide

Reference Books

- Quick Reference Guide
- DMAP Programmer's Guide
- Reference Manual
- Utilities Guide

User's Guides

- Getting Started
- Linear Static Analysis
- Dynamic Analysis
- Embedded Fatigue
- Embedded Vibration Fatigue
- Demonstration Problems
- Thermal Analysis
- Superelements
- Design Sensitivity and Optimization
- Rotordynamics
- Implicit Nonlinear (SOL 600)
- Explicit Nonlinear (SOL 700)
- Aeroelastic Analysis
- User Defined Services
- Non Linear (SOL 400)
- High Performance Computing

You may find any of these documents from MSC Software at:

<http://simcompanion.mscsoftware.com/infocenter/index?page=home>



Technical Support

For technical support phone numbers and contact information, please visit:

<http://www.mscsoftware.com/Contents/Services/Technical-Support/Contact-Technical-Support.aspx>

Support Center (<http://simcompanion.mscsoftware.com>)

The SimCompanion link above gives you access to the wealth of resources for MSC Software products. Here you will find product and support contact information, product documentations, knowledge base articles, product error list, knowledge base articles and SimAcademy Webinars. It is a searchable database which allows you to find articles relevant to your inquiry. Valid MSC customer entitlement and login is required to access the database and documents. It is a single sign-on that gives you access to product documentation for complete list of products from MSC Software, allows you to manage your support cases, and participate in our discussion forums.



Training and Internet Resources

MSC Software (www.mssoftware.com)

MSC Software corporate site with information on the latest events, products and services for the CAD/CAE/CAM marketplace.

<http://simcompanion.mssoftware.com>

The SimCompanion link above gives you access to the wealth of resources for MSC Software products. Here you will find product and support contact information, product documentations, knowledge base articles, product error list, knowledge base articles and SimAcademy Webinars. It is a searchable database which allows you to find articles relevant to your inquiry. Valid MSC customer entitlement and login is required to access the database and documents. It is a single sign-on that gives you access to product documentation for complete list of products from MSC Software, allows you to manage your support cases, and participate in our discussion forums.

<http://www.mssoftware.com/msc-training>

The MSC-Training link above will point you to schedule and description of MSC Seminars. Following courses are recommended for beginning MSC Nastran users.

NAS101A - Linear Static and Normal Modes Analysis using MSC Nastran

This course serves as an introduction to finite element analysis. It includes discussion of basic features available in MSC Nastran for solving structural engineering problems. In this course, all finite element models will be created and edited using a text editor, not a graphical pre-processor. Proper data structure of the MSC Nastran input file is covered. At the conclusion of seminar, the student will be familiar with fundamental usage of MSC Nastran.

NAS101B - Advanced Linear Analysis using MSC Nastran

This course is a continuation of NAS101A - Linear Static and Normal Modes Analysis using MSC Nastran. In this class, you will learn: Theory of buckling analysis and how to perform a buckling analysis About rigid elements - MPC, RBAR,RBE2, and RBE3 Modeling with interface element CINTC and connectors Lamination theory and composite materials MSC Nastran composite theory Failure theories Linear contact and permanent glued contact Different model checks Modeling tips and tricks

NAS120 - Linear Static Analysis using MSC Nastran and Patran

This seminar introduces basic finite element analysis techniques for linear static, normal modes, and buckling analysis of structures using MSC Nastran and Patran. MSC Nastran data structure, the element library, modeling practices, model validation, and guidelines for efficient solutions are discussed and illustrated with examples and workshops. Patran will be an integral part of the examples and workshops and will be used to generate and verify illustrative MSC Nastran models, manage analysis submission requests, and visualize results. This seminar provides the foundation required for intermediate and advanced MSC Nastran applications.



1

Getting Started

- Introduction 2
- Numerical Optimization Basics 9
- Structural Optimization 22
- Summary 26



Introduction

This chapter introduces some of the basic concepts of numerical optimization with an emphasis on the methods used in MSC 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?

What is Design Optimization?

Optimization is a broad and active area technology and we need to first focus the discussion on the particular application that is the subject of this User's Guide. We restrict ourselves to the optimization of structures that can be analyzed with MSC Nastran. In this context, design optimization refers to the search for a structural design that is, in some sense, optimal, or "the best," while varying structural parameters. While performing this search, the design is guided to satisfy operating limits that are imposed on the response of the structure and by further limits on the values the structural parameters can assume. This, in a nutshell, represents the design optimization task in MSC Nastran and leads directly to concepts like "objective," "design variables," "design constraints," "side constraints" and "search", which are covered in this chapter.

What is Design Sensitivity?

The design optimization capability in MSC Nastran benefits significantly from being based on a design sensitivity analysis. Design sensitivity analysis computes the rates of change of structural responses with respect to changes in design parameters. These design parameters, or design variables, can be such things as shell thicknesses, beam dimensions, hole radii, material variables 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 chassis 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.

These sensitivities are computed explicitly in MSC Nastran and are extremely valuable in their own right as they can be used to predict how a design change will affect an important response. When they are applied with an optimizer, they greatly improve the efficiency of the search since the algorithm now not only knows the current state of the design but also has an idea of which way to look for an improved design. MSC, therefore, takes considerable care that the predicted sensitivities are as accurate as possible and applies special techniques to insure that they predict accurate response values as the design variables change.



The basic design optimization capability in MSC Nastran depends on having design sensitivity information available and optimization cannot be performed without it. Design optimization in MSC Nastran is limited to those analyses and design parameters that have been explicitly designated in the design of the capability and that can be accessed by you in the performance of an optimization task. As you will learn from studying this guide, the range of design tasks that MSC Nastran can address is very broad and expands as new versions of the code are released. There will always remain, however, situations that cannot be addressed with MSC Nastran and it is necessary to study this guide to determine if your design task is one that can be treated by the capability.

As a final note to motivate the value of sensitivity analysis, it is useful to consider what would occur if you did not have this information available. If the design task was to find the value of a single design variable that provided the best performance, one could conceive that performing analyses with several values of this parameter would provide information that could be displayed as a function of the variable and the optimum design could be visually determined. Now consider the case with two design variables. The number of analyses required is expanded and the plot now becomes three dimensional, but it still should be possible to find an optimum in this fashion. But now consider 10 variables and then one hundred and then one thousand. At some point, the most powerful computer would be overwhelmed by the number of analyses required and in making sense of the analysis results. With sensitivity analyses, it is quite reasonable to pose design tasks in MSC Nastran with a set of design variables that is far beyond that possible without the sensitivity information. Even for a small number of design variables, the presence of design sensitivities makes finding solutions much more efficient than what would be possible without these sensitivities.

Why Use Design Sensitivity and Optimization?

Why use design sensitivity and optimization? Perhaps you have been requested to investigate product improvement using optimization, or perhaps you are aware that these tools can be used to create better engineered designs. 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 MSC 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.

Example D

Your company supplies automotive components and one of your products is a connecting rod that is produced in the millions for a number of vehicles. The design of the component is well understood and has been refined for years based on detailed design loads. It is now decided to see if shape optimization techniques can be applied to produce a design that satisfies the design requirements at a final weight that is less than the current weight. Even a 1-2% improvement will translate into significant cost savings when it is applied across the large production run.

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 which must be clear in order to make effective use of both.



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 which 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.

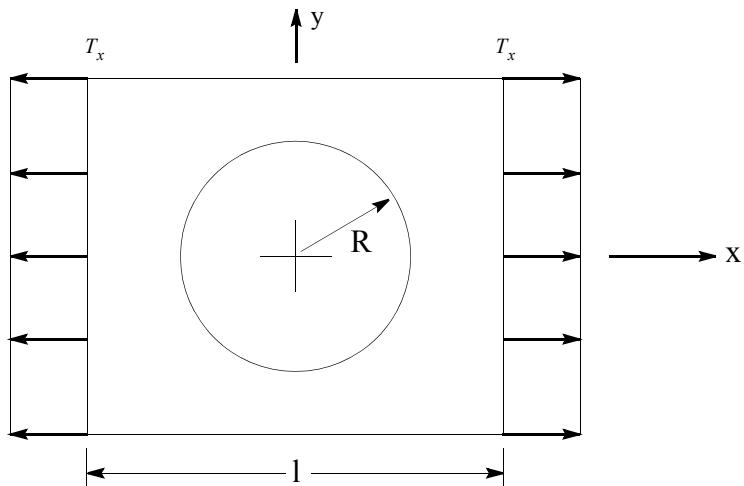


Figure 1-1 Flat Plate with Hole



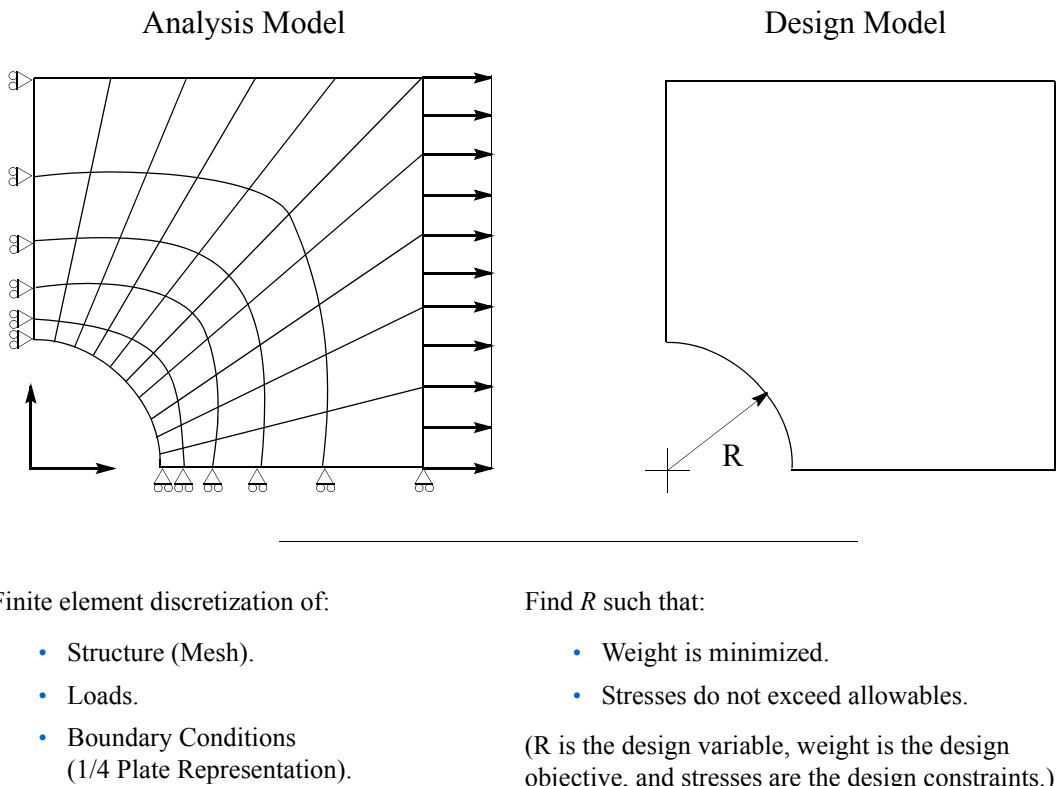


Figure 1-2 Analysis Versus Design Models

In contrast, a design model is an idealized statement of changes which 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. Limits on the range of the design variables are called side constraints. 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 local 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 global optimum or a unique solution (as in the case of analysis).

One thing the analysis and design models share is that the results are dependent upon the skill and judgment 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. In fact, the optimizer can be expected to exploit inaccuracies in the analysis model in a way that helps it perform the design task.

Optimizer Limitations

A numerical optimizer seeks to find an improved design by trying to minimize or maximize a specified 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 which 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 which the optimizer proposes may turn out to be unrealistic. 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 challenge on the factory floor!

What Do I Need to Use Design Optimization Effectively?

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

The topics covered in this user’s guide are intended to describe design sensitivity and optimization in MSC Nastran. No prior knowledge of structural optimization is required. However, it must be recognized that the discussion of structural optimization contained in this guide is restricted to the features appearing in design sensitivity and optimization in MSC Nastran. 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, has enough information available to use it wisely and effectively. More information may, at



times, be desired on a topic than is available is this guide. Two text books that emphasize the concepts of this guide are in References [1](#). and [2](#).

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 truer 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 is more certain. The description of methods for accomplishing these goals forms the majority of the material of this guide.



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.

How Much Do I Need to Know About Optimizers?

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. As an analogy, 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 MSC 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 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 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.

An Illustration

A useful way of introducing basic optimization concepts is to visualize a “design task” of finding the lowest spot in the immediately surrounding terrain. Suppose we are standing on the side of a hill and would like to find the point of lowest elevation; this is our “objective”. We will quantify the location of any point in terms of coordinates and these are our “design variables.” Suppose, also, that some fences exist which 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 local 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 which 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.

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. This will allow us to determine which direction will move us down the hill the fastest and, based on this estimate, 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; that is, we have reached a point at which any move will take us up hill (an unconstrained option) or a point where we can move further downhill only by climbing over a fence (a constrained optimum).

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 as our design variables. 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. The task becomes understandably more complex when considering that an optimizer may have to deal with hundreds of design variables. In fact, recent topology optimization developments have considered one million design variables.

There are three other conditions that may apply to our hill climbing exercise that illustrate two additional concepts. The fences on the hill are properly considered constraints, but we may have been told that under no circumstances can we go north further than a specified amount, even though there is no fence there. This constraint that is imposed directly on the design variable is called a “side constraint.” We might further 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 linear 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 complicated and numerical optimizers are the best way to attack the problem. Another condition is that we may be restricted to coordinates that are a multiple of 10 (10.,20.,30.....1000.). This is the concept of “discrete design variables” and in MSC Nastran this is handled by first finding the continuous optimum and then stepping over to the closest discrete solution that minimizes the elevation.



This illustration is too simple in some ways and we will learn that following the maximum gradient is not always the fastest way to find the bottom of the hill. But we can build on this experience to express the equations which describe the basic optimization problem statement.

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 found in many textbooks.

Find \mathbf{X} to minimize (or maximize)

$$F(\mathbf{X}) \text{ objective} \quad (1-1)$$

subject to

$$g_j(\mathbf{X}) \leq 0 \quad j = 1, \dots, n_g \text{ inequality constraints} \quad (1-2)$$

$$h_k(\mathbf{X}) = 0 \quad k = 1, \dots, n_h \text{ equality constraints} \quad (1-3)$$

$$x_i^L \leq x_i \leq x_i^U \quad i = 1, \dots, n \text{ side constraints} \quad (1-4)$$

$$\mathbf{X} = \{x_1, x_2, \dots, x_n\} \text{ design variables} \quad (1-5)$$

In this notation, items that are in upper case and bold are vectors while members of the vectors are designated using a lower case symbol with a subscript to indicate the member.

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 are expressed in a less than or equal to zero form by convention; that is, a constraint is satisfied if its value is negative. The location of the j -th “fence” lies at $g_j(\mathbf{X}) = 0$. Equality constraints, if present, must be satisfied exactly at the optimal design.

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 which can deal with this nonlinearity must be used. MSC Nastran includes capabilities for solving both linear and nonlinear optimization problems, with nonlinear problems making up the vast majority of the prescribed tasks.

As seen throughout the remainder of this user’s guide, the basic optimization problem statement is used directly in MSC 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 [Developing the Design Model 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.

Find x_1 and x_2 to minimize the objective

$$F(\mathbf{X}) = x_1 + x_2 \quad (1-6)$$

subject to the constraints

$$g_1(\mathbf{X}) = \frac{1}{x_1} + \frac{1}{x_2} - 2 \leq 0 \quad (1-7)$$

$$x_1 \geq 0.1 \quad x_2 \geq 0.1 \quad (1-8)$$

The objective and constraint functions are dependent on two design variables x_1 and x_2 . The objective $F(\mathbf{X})$ is linear in the design variables, while the constraint $g_1(\mathbf{X})$ 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 .



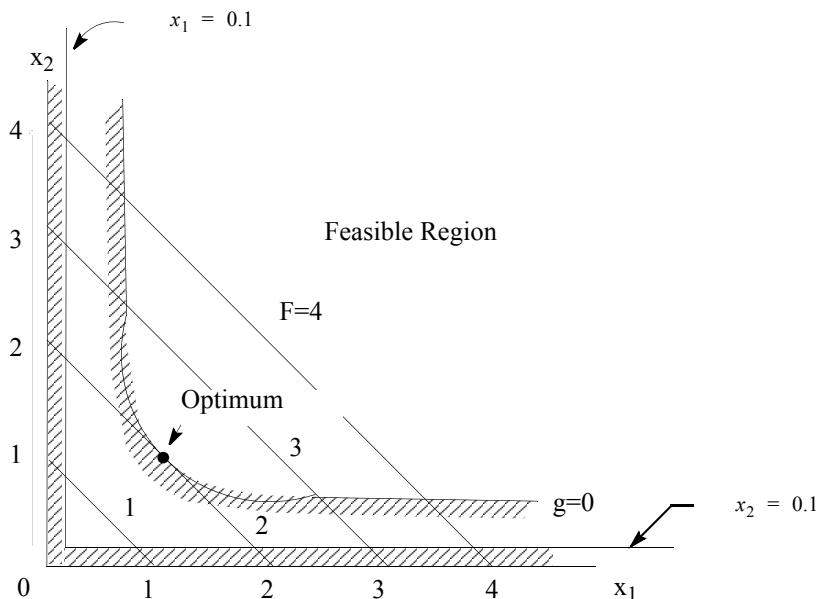


Figure 1-3 Two-Variable Function Space

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.

Numerically Searching for an Optimum

The optimization algorithms in MSC 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. (One exception to the gradient based optimizers is the [Fully Stressed Design](#)).

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 search direction. 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 MSC 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.

The first step in a numerical search procedure is to determine 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

$$\frac{df(x)}{dx} \approx \frac{f(x + \Delta x) - f(x)}{\Delta x} \quad (1-9)$$

where the quantity Δx 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 gradient vector of partial derivatives of the function can be written as

$$\nabla F(\mathbf{X}) = \begin{Bmatrix} \frac{\partial F}{\partial x_1} \\ \vdots \\ \frac{\partial F}{\partial x_n} \end{Bmatrix} \approx \begin{Bmatrix} \frac{F(\mathbf{X} + \Delta x_1) - F(\mathbf{X})}{\Delta x_1} \\ \vdots \\ \frac{F(\mathbf{X} + \Delta x_n) - F(\mathbf{X})}{\Delta x_n} \end{Bmatrix} \quad (1-10)$$

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

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

$$\mathbf{S} = -\nabla F \quad (1-11)$$

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

For now, just note that MSC 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 direction, subsequent search directions often fail to improve the objective function significantly. In MSC Nastran we use other, more efficient methods which 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?

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 \mathbf{S} and a vector of design variables \mathbf{X} , the new design at the conclusion of our search in this direction can be written as



$$\mathbf{x}^1 = \mathbf{x}^0 + \alpha^* \mathbf{s}^1 \quad (1-12)$$

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 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

$$F^1 = F(\mathbf{x}^0 + \alpha^* \mathbf{s}^1) \quad (1-13)$$

$$g_j^1 = g_j(\mathbf{x}^0 + \alpha^* \mathbf{s}^1) \quad j = 1, \dots, n_g \quad (1-14)$$

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 which 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 criterion to see if it is met and an optimum has been found. This required definition is provided by the Kuhn-Tucker conditions (see [The IPOPT Algorithm](#)), which are physically quite intuitive.

[Figure 1-4](#) shows a two design variable space with constraints $g_1(\mathbf{x})$ and $g_2(\mathbf{x})$ and objective function $F(\mathbf{x})$. 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 which lies at the intersection of the two constraints. This location is shown as \mathbf{x}^* .



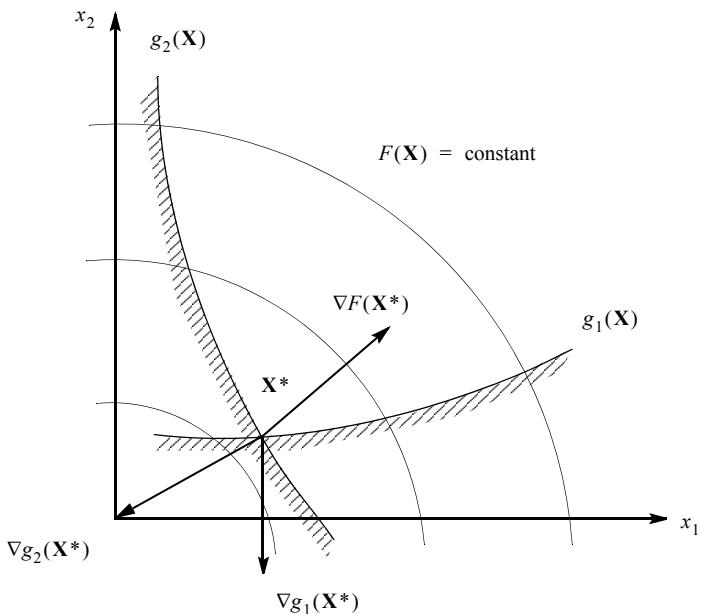


Figure 1-4 Kuhn-Tucker Condition at a Constrained Optimum

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 which 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 which enable the zero vector sum condition to be met. We can convince ourselves that this condition could not be met for any other point in the neighboring design space.



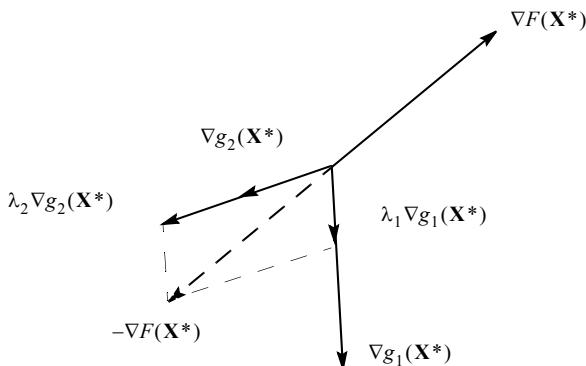


Figure 1-5 Graphical Interpretation of Kuhn-Tucker Conditions

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 MSC Nastran tests the Kuhn-Tucker conditions in connection with the search direction determination algorithm. See [The IPOPT Algorithm](#) for theoretical details.

A Simple Structural Example

Let's take a look at a simple example to help illustrate optimization concepts and to introduce some qualitative aspects of the optimizer used in MSC 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.



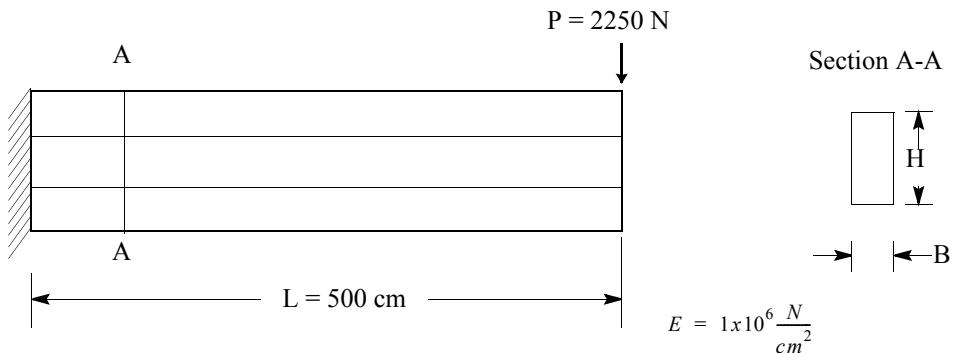


Figure 1-6 Cantilever Beam

The optimization problem statement could be written as follows

minimize

$$V = B \cdot H \cdot L \quad (1-15)$$

subject

$$\sigma = \frac{Mc}{I} = \frac{6PL}{BH^2} \leq 700 \frac{\text{N}}{\text{cm}^2} \quad (1-16)$$

$$\delta = \frac{PL^3}{3EI} = \frac{4PL^3}{BH^3E} \leq 2.54 \text{ cm} \quad (1-17)$$

$$\frac{H}{B} \leq 12 \quad (1-18)$$

$$1 \leq B \leq 20 \quad (1-19)$$

$$20 \leq H \leq 50 \quad (1-20)$$

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



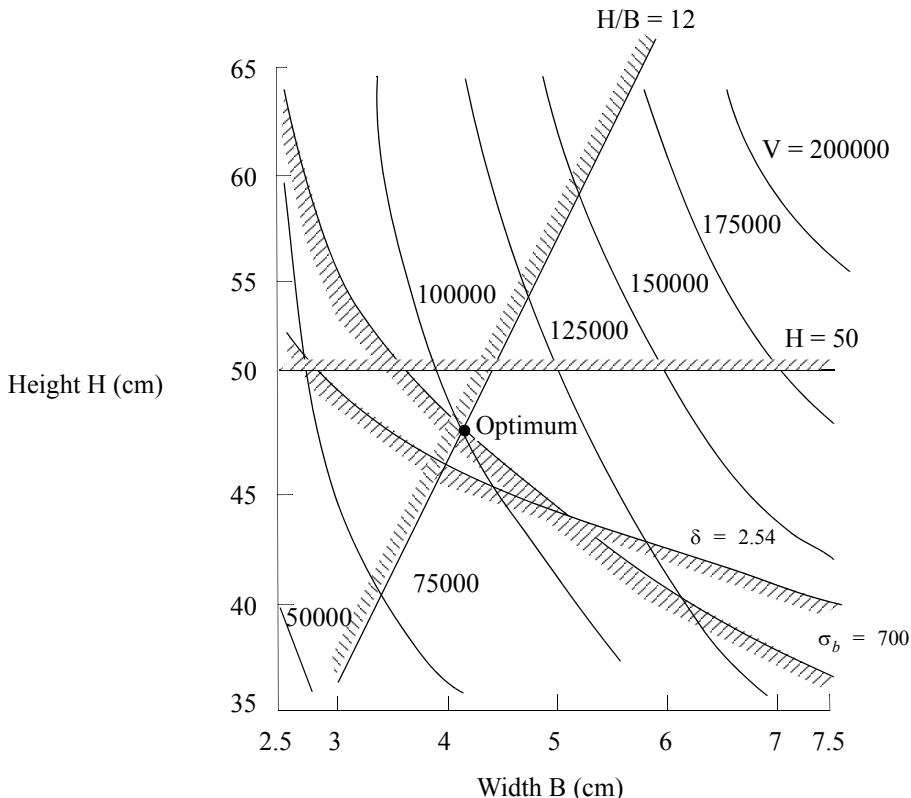


Figure 1-7 Cantilever Beam Design Space

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 $100,000 \text{ cm}^3$. Let us examine the path which the optimizer might take as it searches for this constrained minimum by referring to [Figure 1-8](#).

Assume that we begin with an initial design of $H = 44 \text{ cm}$ and $B = 7 \text{ cm}$. 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.



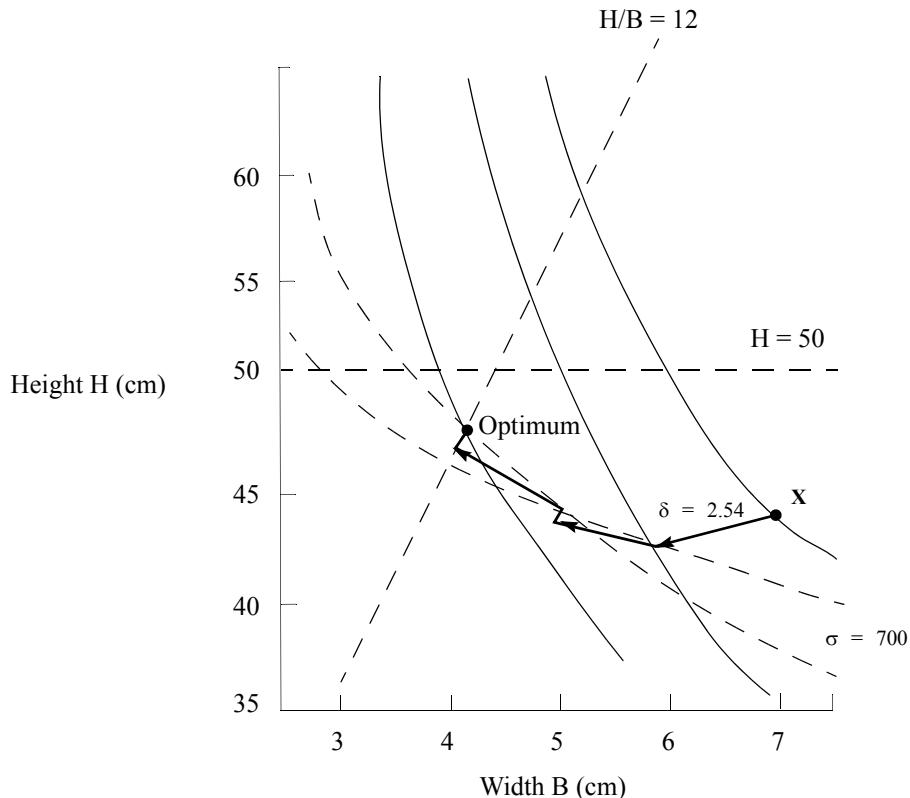


Figure 1-8 Sequence of Iterations: Modified Method of Feasible Directions

The optimizer in MSC 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.

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 MSC 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.

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.

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.

We can also preview a concept that we will deal with in great detail in [Fundamentals of Design Sensitivity and Optimization in MSC Nastran](#); optimizers work best when they are dealing with linear problems. MSC Nastran applies concepts that make the structural response quantities a nearly linear function of the design variables (see [The Approximate Model](#)). The constraint contained in [Equation \(1-18\)](#) is highly nonlinear the way it is written and the representation of [Figure 1-7](#) suggests that this can be linearized as

$$H - 12B \leq 0 \quad (1-21)$$

In practice, this linear specification of the constraint between the two design variables is a much easier constraint to deal with than the non-linear one.

In summary, the intent of this example is 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 as implemented in MSC Nastran 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 a topic of the next section.



Structural Optimization

[Introduction](#) and [Numerical Optimization Basics](#) 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. This final introductory section focuses more closely on how the optimization methods are coupled with a structural analysis in general and with the MSC Nastran product in particular. This is done by first introducing the concepts of design properties and design responses that link the features of MSC Nastran that are familiar to the analyst to the design variables and constraints that are familiar to optimizer. Following paragraphs address the difficulties encountered when trying to make the optimization process feasible for “real-world” design tasks. Optimization 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. Methods used in MSC Nastran to reduce the number of finite element analyses required are introduced, as are methods for minimizing the effort involved in providing the information required during the optimization.

Relating the Finite Element Analysis Model to the Design Model

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. The properties define materials, element thicknesses, element meshes and other quantities needed to perform the analysis. The results of the analyses depend on the type of analysis performed, but include such things as element stresses and modal frequencies. In design modeling, on the other hand, we wish to define how our model can change in pursuit of an improved design along with the criteria which we (or the optimizer) use to judge the effects of these changes. [Numerical Optimization Basics](#) has presented the basics of the optimization task and uses words such as objective, constraints and design variables. MSC Nastran links these two models by introducing two intermediate concepts.

The first is the designed property and provides a link between the design variable used by the optimizer and the finite element property used in the finite element analysis. The second concept is the design response. A design response is a physically based result that can be used as the design objective or it can be used, with user imposed limits, as a design constraint. The presence of this intermediate layer between the design quantities and the analysis quantities is a key feature of Design Sensitivity and Optimization that adds to the flexibility of the capability. Design properties and design responses are treated in detail in [Developing the Design Model for Sensitivity and Optimization](#).

Managing the Structural Optimization Task

Historically, the first attempts at linking structural analysis with numerical optimization were based on 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. One must also remember that these methods were being developed during the infancy of computerized structural analysis when a finite element analysis that we now consider trivial consumed a measurable



amount of computer resources. Therefore, the sheer number of analyses quickly tended to make this approach impractical in all but the smallest of problems. Recall that the numerical optimizer may not only require 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. MSC Nastran therefore employs concepts that limit the number of required finite element analyses. It is remarked that some commercial structural analysis codes that include an optimization capability still perform their optimization tasks in the “black box” manner of [Figure 1-9](#). Although they benefit from the high performance capabilities of present day computers, they are still limited in the size of the design task (in terms of number of design variables and responses and the cost of finite element analysis) they can address relative to what can be achieved using MSC Nastran.

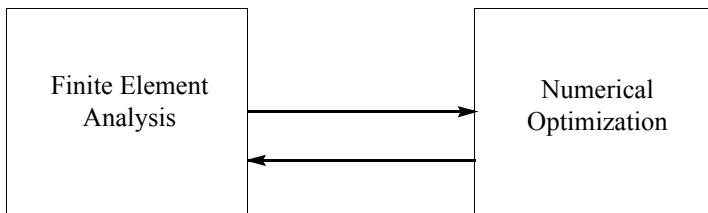


Figure 1-9 Early Structural Optimization Attempts

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 this section introduces each of these methods, all of which are available in MSC 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.

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. 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.”

The next step might then be to identify the few constraints that are violated or nearly violated. There may be just a few constraints which 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. In structural optimization, this is 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 (sensitivity analysis costs are reduced as well).

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 make use of the results of the sensitivity analysis to construct an approximation to the true design space which, although only locally valid, is explicit in the design variables. The resultant explicit representation can then be used by the optimizer whenever function or gradient 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 which 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 MSC Nastran.



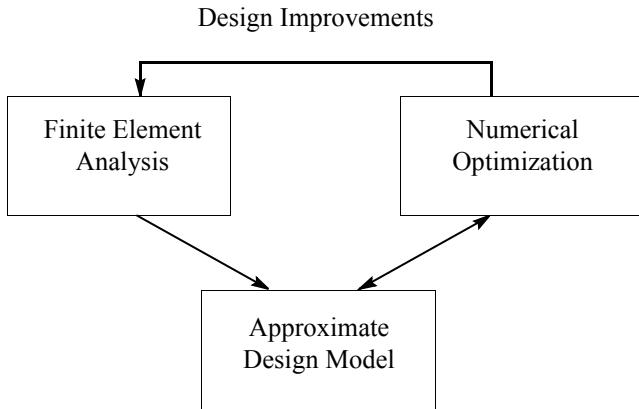


Figure 1-10 Coupling Analysis and Optimization Using Approximations

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 actually managed to satisfy the various design constraints and reduce the objective function. This reanalysis update of the proposed designs 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.



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.



2

Fundamentals of Design Sensitivity and Optimization in MSC Nastran

- Overview of Fundamentals 28
- Types of Optimization 32
- The Design Model 33
- Multidisciplinary Analysis 43
- Constraint Screening 47
- Design Sensitivity Analysis 50
- Optimization 74
- The Approximate Model 78
- Tests for Convergence 98



Overview of Fundamentals

Chapter 1 presented a Getting Started overview of design optimization in general and introduced a number of concepts that are specific to MSC Nastran's implementation of Design Sensitivity and Optimization. This chapter goes into greater depth on the MSC Nastran methodology, again in terms of concepts and equations. Mention of user interface features (that is, the case control commands and bulk data entries) is sometimes unavoidable, but the intent is to defer a description of the interface to [Developing the Design Model for Sensitivity and Optimization](#). This chapter has been prepared for several classes of readers:

- New users who want to obtain an overview of features of MSC Nastran's implementation of Design Sensitivity and Analysis.
- More experienced users who will use this chapter as reference material to clarify details of the features and to gain insight into the algorithms employed.
- Support engineers and software developers who need to be able to explain, and possibly extend, the capability.

It is recognized that this covers a broad spectrum of readers and you probably will not read this chapter from beginning to end. Instead, you should skim the material until you find the topic that interests you and return to the chapter as you gain more experience and want to obtain a deeper level understanding.

The majority of this chapter can be thought of as a description of the flow chart shown in [Figure 2-1](#), which is itself an expanded version of the sketch of [Figure 1-10](#). The figure shows a design loop with a number of blocks. The blocks, and the section in which they are described, are as follows:



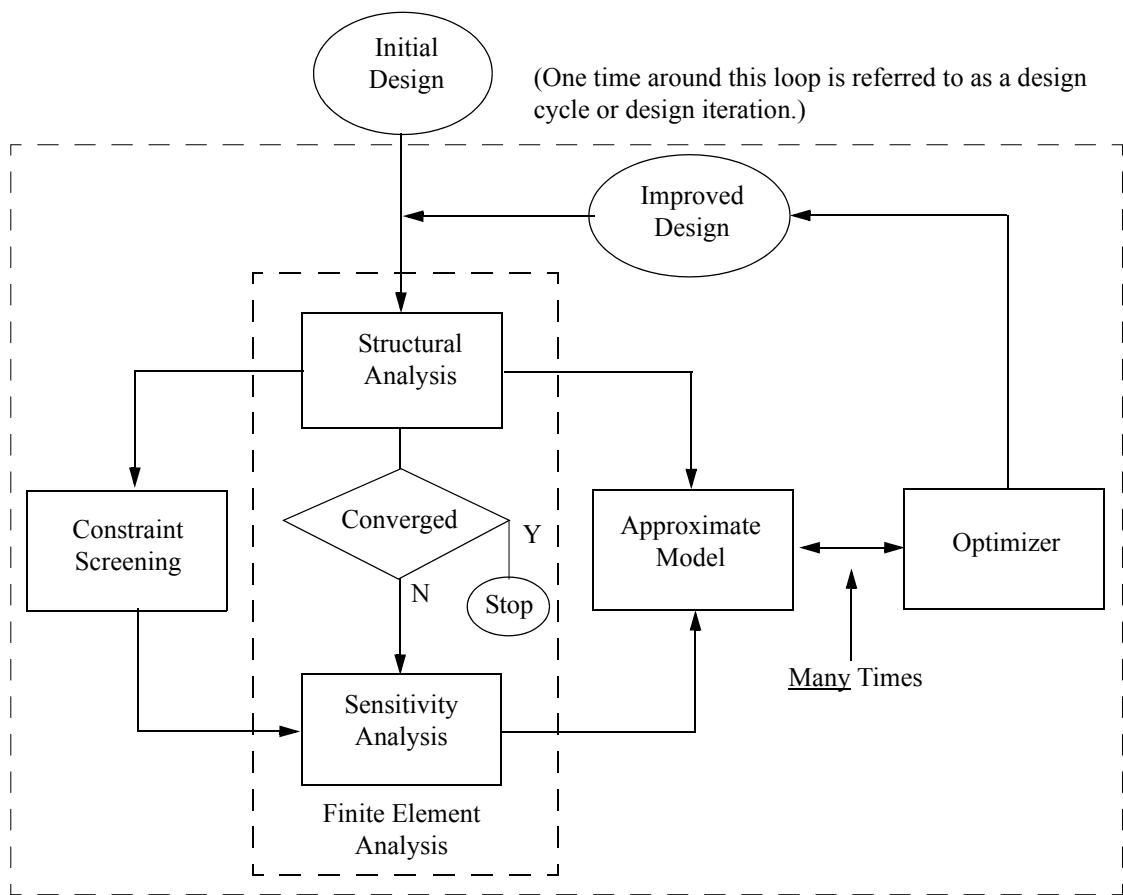


Figure 2-1 MSC Nastran Implementation of Structural Optimization

Initial Design

The initial design is a combination of the analysis model familiar to MSC Nastran users and the design model, which is a unique feature of the Design Sensitivity and Optimization capability. The design model is discussed in [The Design Model](#).

Structural Analysis

This is the analysis function that is, again, familiar to the MSC Nastran user. However, for design sensitivity and optimization, it is frequently necessary to perform multiple types of analyses. This reflects the fact that the design responses can be created by a number of analysis types and it is necessary for the



optimizer to synthesize these results when performing its redesign. [Multidisciplinary Analysis](#) discusses this “multidisciplinary” feature.

Constraint Screening

This concept was introduced in [Structural Optimization](#) and refers to the process that is used to identify those constraints that are likely to drive the redesign process. A detailed discussion on this is presented in [Constraint Screening](#).

Sensitivity Analysis

Accurate sensitivity analyses are the hallmark of the MSC Nastran implementation of design sensitivity and optimization. The individual analysis types each have their own techniques for performing sensitivity analysis and these are developed in [Design Sensitivity Analysis](#).

Optimizer

MSC Nastran uses a variety of optimization algorithms from the MSCADS suite of algorithms as well as the IPOPT algorithm.

The MSCADS algorithms are adapted from the public domain versions of ADS codes described in Reference 3. An extensive discussion of the theory contained in these algorithms can be found in Reference 1. The IPOPT algorithm is a special purpose optimization algorithm to address design tasks with a large number (> 3000 to 4000) of design variables and has its primary application to topology optimization tasks, but can be applied to the conventional design tasks that are discussed in this document as well. The IPOPT algorithm is open source code available from COIN-OR and maintained by IBM. It implements an interior point line search filter method, as described in Appendix C. The DOT and BIGDOT algorithms from Vanderplaats R&D Inc. are not available in Nastran 2010 and subsequent releases. However, the DOT and BIGDOT algorithms can be found in earlier releases.

Approximate Model

[Structural Optimization](#) has also introduced this concept that involves the construction of high-quality approximations to the finite element results so that the number of full scale finite element analyses is kept to a minimum. [The Approximate Model](#) documents the approximation concepts used in MSC Nastran.

The Improved Design

This is the point at which the finite element model is updated based on the results from the optimizer so that a new finite element analysis can occur. This is a straightforward operation and its discussion is included in the Converged write-up of [Tests for Convergence](#).



Converged

As [Figure 2-1](#) indicates, Design Optimization is an iterative process and a key part of the implementation is therefore determining when to stop the iterations. [Tests for Convergence](#) discusses the many factors that enter into making this decision.



Types of Optimization

MSC Nastran performs three basic types of optimization that differ primarily in how the relationship between the design task and the analysis parameters is specified. For now, these types are introduced and the remainder of the guide will go into greater detail on each of the types.

Sizing Optimization

Sizing optimization refers to a design task where the analysis quantities that can vary are called out explicitly using DVxRELY entries that permit the design of such items as plate thickness, Young's modulus and spring stiffnesses (see Design Model)

Shape Optimization

For shape optimization, the design variables affect the locations of the grids that make up the finite element model. The optimization task is then one which seeks the best shape to meet the prescribed objective and constraints.

Toptimization

“Toptimization” is a MSC created word that lumps three types of redesign that share the feature that simple user inputs spawn internal design variables that lead to innovative designs that would not be achievable using conventional design techniques. These techniques are discussed in detail in Chapter 7 (Topology, Topometry and Topography Optimization) and are mentioned here simply to set the stage.

Topology Optimization

Topology optimization can be thought of as a way to develop design concepts starting from a very basic specification of the design task in terms of loads and boundary conditions and the design space. The algorithm determines the subset of the material space that provides the optimum behavior. In this case, a determination is made whether each designable finite element should be there or not.

Topometry Optimization

Topometry optimization provides a simple way of generating a design task that permits each designated element to be separately designed. For example, if 1000 elements share a property ID, a single user input can spawn an individual design variable and property id for each of these elements.

Topography Optimization

Topography optimization is a special class of shape optimization where the identified finite elements grids can move normal to the shell surface. It can be quite powerful in improving the response of sheet metal parts with a negligible increase in weight.



The Design Model

[Numerical Optimization Basics](#) introduced the general concepts of design variables, constraints, objective function, and side constraints. [Structural Optimization](#) presented further concepts of designed properties and design responses and indicated that these intermediate quantities are used to fill the gap between standard finite element analysis and optimization procedures. This section of the guide indicates how these concepts are implemented in MSC Nastran.

Design Variables

Design Variables are quantities that are known to the optimizer and that can be directly changed to satisfy the optimization statement. In MSC Nastran, the design variables are defined on the DESVAR entry (see [Defining the Design Variables](#) and [Bulk Data Entries](#)) and can only affect the finite element analysis if they are used with designed properties, shapes and/or design responses. It is the user's responsibility to define the initial values of these design variables and, optionally, their upper and lower bounds. The bounds make up the "side constraints" (See [Equation \(1-4\)](#)) of the optimization problem statement and these bounds can never be exceeded at any time.

MSC Nastran distinguishes between dependent and independent design variables. Dependent design variables are linked to the independent variables using a relationship of the form

$$x_j^D = c_0 + \sum_{i=1}^{ndv} c_i x_i^I \quad (2-1)$$

MSC Nastran also supports the concept of "discrete" design variables wherein the value of the design variable is restricted to a user specified set of real numbers. This topic is discussed in more detail as part of [Discrete Variable Optimization](#) (Ch. 9).

Designed Properties

Designed properties are quantities that do have a direct bearing on the finite element analysis and include such things as element thicknesses, material properties, mass values, and grid locations. Designed grid locations, which are used in shape optimization, are defined in a way that is distinct from the remaining properties. Therefore, the link between grid locations and design variables is presented separately in [Designed Shapes](#) that follows immediately after this discussion which deals with element properties, material properties, connectivity properties and load properties.

The first order of business is to explicitly define what these properties are. Element properties refer to quantities defined in MSC Nastran using element property bulk data entries (those that begin with the letter 'P'). [Table 2-1](#) contains a list of the bulk data entries that have element properties that can be designed and identifies which of the fields on the entry can be designed. In general, real numbers that appear on an element property bulk data can be designed. In Table 2-1, property type GPLY refers to a particular ply ID that appears on one or more PCOMPG entries. It enables the design of the thickness or orientation angle for this ply ID across all PCOMPG entries that reference it.



Table 2-1 Element Properties Available as Designed Properties

Property Entry	Property
PACABS	TESTAR, CUTFR, B, K, M
PACBAR	MBACK, MSEPTM, FRESON, KRESOM
PBAR	A, I1, I2, J, NSM, C1, C2, D1, D2, E1, D2, F1, F2, K1, K2, I12
PBARTL	DIMi, NSM
PBEAM	(A(i), I1(i), I2(i), I12(i), J(i), NSM(i), C1(i), C2(i), D1(I), D2(i), E1(i), E2(i), F1(i), F2(i), i = A, B, 1 ... 9), K1, K2, S1, S2, (NSI(j), CW(j), M1(j), M2(j), N1(j), N2(j), j = A, B)
PBEAML	(DIMi(j), NSM(j), j = A, B, 1 ... 9)
PBRSECT/PBMSECT	W (Overall Width), H (Overall Height), T (Overall Thickness) and T(n) Thickness of segment. See PBRSECT, 661 for more information.
PBUSH	(Ki, Bi, GEi, i = 1, 6), SA, ST, EA, ET
PBUSH1D	K, C, M, SA, SE (only linear elements are supported for optimization)
PCOMP	Z0, NSM, SB, TREF, GE, Ti, THETAi
PDAMP	B1, B2, B3, B4
PELAS	K1, GE1, S1, K2, GE2, S2
PGAP	U0, F0, KA, KB, KT, MU1, MU2
PCOMPG	Z0, NSM, SB, TREF, GE, Ti, THETAi
PMASS	M1, M2, M3, M4
PROD	A, J, C, NSM
PSHEAR	T, NSM, F1, F2
PSHELL	T, 12I/T**3, TS/T, NSM, Z1, Z2 (The 12I/T**3 term can be designed but must be referenced by Field ID = 6 rather than by name.)
PTUBE	OD, T, NSM
PVISC	CE1, CR1, CE2, CR2
GPLY	T, THETA

Table 2-2 contains a list of the bulk data entries for material properties and again identifies which fields on these entries can be designed. **Table 2-3** performs the same function for connectivity entries (bulk data entries that begin with the letter 'C'). It is seen that this enables the support of properties such as beam offset vectors, concentrated mass values, and tapered shell elements. **Table 2-4** indicates which external loads properties can be designed in a static analysis. In addition to the properties of these four tables, a final design quantity is the deflection of a control surface (see entry DVPSURF).



Table 2-2 Material Properties Available as Designed Properties

Material Entry	Material Properties
MAT1	E, G, NU, RHO, A, TREF, GE
MAT2	G11, G12, G13, G22, G23, G33, RHO, A1, A2, A3, TREF, GE
MAT3	EX, ETH, EZ, NUXTH, NUZTH, NUZX, RHO, GZX, AX, ATH, AZ, TREF, GE
MAT8	E1, E2, NU12, G12, G1Z, G2Z, RHO, A1, A2, TREF, Xt, Xc, Yt, Yc, S, GE
MAT9	G11, G12, G13, G14, G15, G16, G22, G23, G24, G25, G26, G33, G34, G35, G36, G44, G45, G46, G55, G56, G66, RHO, A1, A2, A3, A4, A5, A6, TREF, GE
MAT10	BULK, RHO, C, GE

Table 2-3 Connectivity Properties Available as Designed Properties

Connectivity Entries	Connectivity Properties
CBAR	X1, X2, X3, W1A, W2A, W3A, W1B, W2B, W3B
CBEAM	X1, X2, X3, BIT, W1A, W2A, W3A, W1B, W2B, W3B
CBUSH	X1, X2, X3, S, S1, S2, S3
CDAMP2,4	B
CELAS2	K, GE, S
CELAS4	K
CGAP	X1, X2, X3
CMASS2,4	M
CONM1	M11, M21, M22, M31, M32, M33, M41, M42, M43, M44, M51, M52, M53, M54, M55, M61, M62, M63, M64, M65, M66
CONM2	M, X1, X2, X3, I11, I21, I22, I31, I32, I33
CONROD	A, J, C, NSM
CQUAD4,8,R	THETA, ZOFFS, T1, T2, T3, T4
CTRIA3,5,R	THETA, ZOFFS, T1, T2, T3
CTRIAX6	TH

Table 2-4 External Load Properties Available as Designed Properties

Loads Entry	Load Property
FORCE	F,N1,N2,N3
MOMENT	M,N1,N2,N3
LOAD	S,Si

Type-1 Properties

Two methods are provided to relate the properties of [Table 2-1](#) through [Table 2-4](#) to the DESVAR values. The first, which is designated type-1, specifies a linear relationship which can be expressed as

$$p_j = c_o + \sum_i c_i x_i \quad (2-2)$$

where p_j is the j-th property, expressed as a linear combination of n design variables plus an optional constant value. In a typical application, the property is a function of a single design variable, but [Equation \(2-1\)](#) 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 the plate thickness distribution of the simple structure shown in [Figure 2-2](#) is to be determined using a combination of constant, linear, and quadratic basis functions as shown in [Figure 2-3](#). The design variables act as multipliers of these basis functions which are, in turn, used to specify the element plate thicknesses.

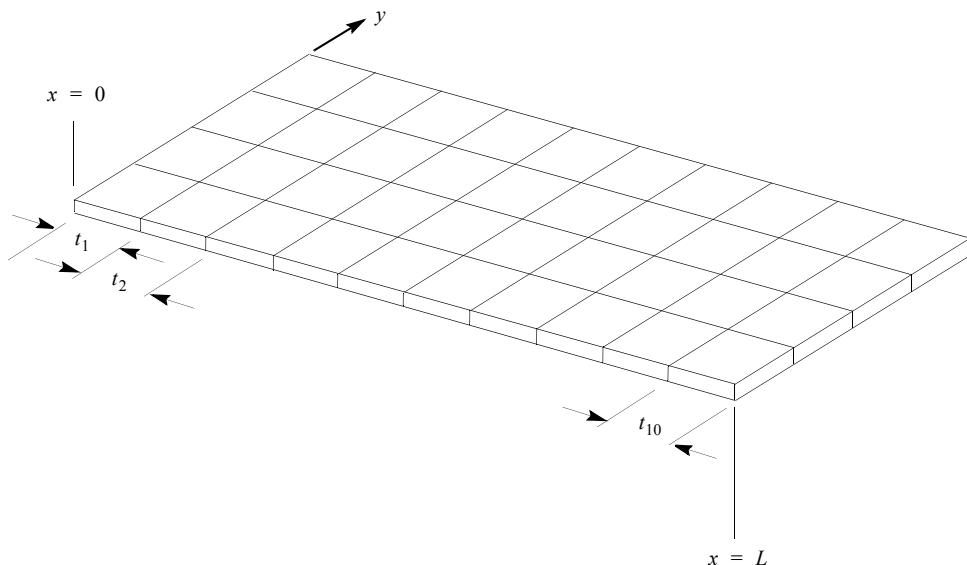
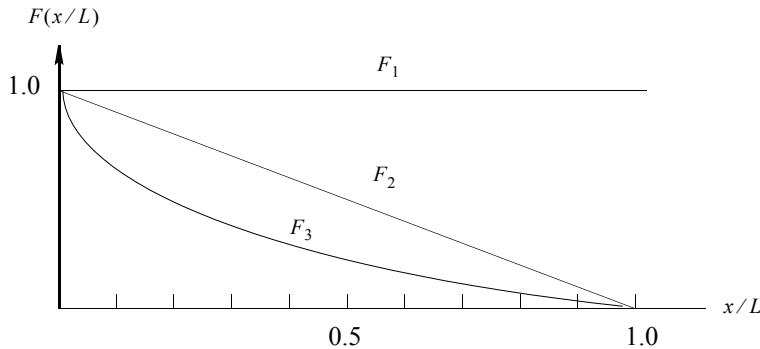


Figure 2-2 Plate Thickness Distribution





$$F_1\left(\frac{x}{L}\right) = 1.0$$

$$F_2\left(\frac{x}{L}\right) = c_1\left(1 - \frac{x}{L}\right)$$

$$F_3\left(\frac{x}{L}\right) = c_1\left(1 - \frac{x}{L}\right)^2$$

Figure 2-3 Basis Functions

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

$$\begin{Bmatrix} t_1 \\ t_2 \\ t_3 \\ t_4 \\ t_5 \\ t_6 \\ t_7 \\ t_8 \\ t_9 \\ t_{10} \end{Bmatrix} = \begin{bmatrix} 1.0 & 1.0 & 1.0 \\ 1.0 & 0.9 & 0.81 \\ 1.0 & 0.8 & 0.64 \\ 1.0 & 0.7 & 0.49 \\ 1.0 & 0.6 & 0.36 \\ 1.0 & 0.5 & 0.25 \\ 1.0 & 0.4 & 0.16 \\ 1.0 & 0.3 & 0.09 \\ 1.0 & 0.2 & 0.04 \\ 1.0 & 0.1 & 0.01 \end{bmatrix} \begin{Bmatrix} x_1 \\ x_2 \\ x_3 \end{Bmatrix} \quad (2-3)$$

The thicknesses of the ten element groups are controlled using only three design variables. Each of the matrix columns correspond to constant, linear, and quadratic basis functions, respectively. Relations of this sort are termed reduced basis formulations.



In general, a reduced basis formulation can be expressed as

$$\{P\}_M = [T]_{MXN}\{x\}_N \quad (2-4)$$

The most powerful form of [Equation \(2-4\)](#) occurs when the number of rows of $[T]$ is much greater than the number of columns or $M \gg N$. [Cantilevered Plate](#) presents an example that uses this concept in the design of a cantilevered plate.

Type-2 Properties

The second method, designated type-2, utilizes a general equation utility to define the relationship between design variables and designed properties

$$p_j = f(\mathbf{X}, \mathbf{C}) \quad (2-5)$$

where the j -th property is a function of a collection of design variables and constants as indicated by the bold notation.

A simple example of the application of a type-2 relation is in the design of beam cross-sectional properties. For a rectangular cross section, the logical design variables would be the width and height of the beam but MSC Nastran expects cross-sectional properties such as area and moment of inertia. If x_1 represents the width and x_2 represents the height, these properties can be written using equations

$$\begin{aligned} A &= x_1 \cdot x_2 \\ I &= \frac{x_1 \cdot x_2^3}{12} \end{aligned} \quad (2-6)$$

It should be noted that the beam library feature of MSC Nastran permits the direct design of the PBARL and PBEAML dimensions. This is a much simpler user interface and is to be preferred if the beam library supports the cross section type of interest.

Type-3 Properties

The beam library does introduce a third type of designed property that is somewhat hidden from the user. The user has direct access to the beam dimension in terms of designed properties. The actual beam properties, such as area and moment of inertia, that are used in the development of the element stiffness matrices, are functions of these dimensions. When a beam property is designed indirectly because it is a function of a beam dimension, the designed beam property is referred to as a “spawned” design property. The relation between the spawned property and the design variable can be linear (stress recovery points are typically a linear function of the beam dimensions) or nonlinear (inertia properties are typically a nonlinear function of the beam dimensions). MSC Nastran makes a special effort to determine the nature of this relationship because the sensitivity calculations are much simpler when the relationship is linear. Spawns beam properties that are spawned using nonlinear relations are referred to subsequently as type-3 properties.



Designed Shapes

Shape sensitivity and optimization in MSC Nastran requires the definition of design variables and a relationship between these variables and the allowable shape variations. The shape variations are similar to the basis vectors of the previous section in that a single shape typically affects multiple grid locations. The amount the design variable is changed during optimization results in a corresponding shape change.

The allowable shape changes are defined using shape basis vectors. The optimizer then determines how much the structure can change by modifying the design variables.

The various methods supported by MSC Nastran for the generation of basis vectors are described in [Shape Optimization](#).

Design Responses

Design responses are used in MSC Nastran as the basis for defining the design objective and design constraints. In a manner similar to the designed properties, they are an intermediate layer that serves to connect the MSC Nastran analysis with the Basic Optimization Problem Statement. [Designating the Design Responses](#) goes into the specifics of the available response quantities so that this discussion simply introduces the three types of responses.

Type-1 Responses

The first type of response is entitled type-1 or first-level responses. These responses are available directly from an MSC Nastran analysis. Structural weight, displacements at grid points, element stresses, and so on, are all examples of type-1 responses. The DRESP1 Bulk Data entry is used to select this type of response and is discussed at length in [Bulk Data Entries](#).

Type-2 Responses

The second type is entitled type-2, or second-level responses. This class of responses is also frequently called user-defined since they utilize the equation input feature in MSC Nastran. This response allows the user to develop a response of the form

$$r_j^2 = f(\mathbf{X}, \mathbf{C}, \mathbf{R1}, \mathbf{P}, \mathbf{R2}, \mathbf{XYZ}) \quad (2-7)$$

where the \mathbf{X} , \mathbf{C} , $\mathbf{R1}$, \mathbf{P} , $\mathbf{R2}$ and \mathbf{XYZ} vectors are design variables, constants, type-1 responses, designed properties, type-2 responses and grid locations, respectively. This is an extremely powerful feature of the MSC Nastran Design Sensitivity and Optimization capability that allows users to synthesize a wide variety of design conditions that are not available from the MSC Nastran analysis directly. Example applications of type-2 responses are to formulate stress failure criteria not available in MSC Nastran, to evaluate local buckling behavior, to develop root mean square responses and, in shape optimization, to impose limits that prevent the geometry from becoming physically meaningless (e.g., to make sure a hole radius does not extend beyond the boundary of the plate it is contained in). The DRESP2 Bulk Data entry is used to develop this type of response.



Type-3 Responses

The final type of response is called type-3 or third level responses. This can be regarded as a major extension of the type-2 response that creates a response using a representation similar to that of [Equation \(2-7\)](#)

$$r_j^3 = f(\mathbf{X}, \mathbf{C}, \mathbf{R1}, \mathbf{P}, \mathbf{R2}, \mathbf{XYZ}, \text{string}) \quad (2-8)$$

The evaluation of the type-3 response is performed using an API (Application Programming Interface) that invokes a process running externally to MSC Nastran. The additional ‘string’ argument in the type-3 formulation permits the passing of general information that can be used to direct the execution of the external process. For example, it could identify an input file needed by the external evaluator or it could be a simple flag that directs the execution of the external evaluator in some way. It is seen that the type-3 response provides a completely general way of synthesizing response quantities that can drive the design but which are not MSC Nastran responses and are too complex to be evaluated using the type-2 formulation. Examples are limited only by the imagination and cleverness of the user, and include proprietary buckling criteria, cost models, detailed component analysis and other analyses not performed in MSC Nastran. The DRESP3 Bulk Data entry is used to develop this type of response.

A complete description of Type-3 Responses is given in [External Response](#).

Design Objective

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

Design Constraints

Design constraints are invoked by using the DCONSTR entry to identify a design response (DRESP1, DRESP2 or DRESP3) and to impose limits on the response

$$r_j^L \leq r_j(\mathbf{X}) \leq r_j^U \quad (2-9)$$

where r_j^L is the lower bound on the j -th response and r_j^U is the corresponding upper bound. Unlike the design objective, constraints can be applied to responses that have multiple values. For example, a stress response that invokes a particular property ID will create a separate response value for every finite element that references that property ID.

Normalized Constraints

The inequalities of [Equation \(2-9\)](#) are converted to normalized constraints that satisfy the convention of [Equation \(1-2\)](#) in which satisfied constraints are negative



$$\begin{aligned} g_{2j-1}(\mathbf{X}) &= \frac{r_j^L - r_j(\mathbf{X})}{GNORM} \leq 0 \\ g_{2j}(\mathbf{X}) &= \frac{r_j(\mathbf{X}) - r_j^U}{GNORM} \leq 0 \end{aligned} \quad (2-10)$$

where

$$GNORM = \begin{cases} |\text{LALLOW}| \text{ for lower bounds if } |\text{LALLOW}| > \text{GSCAL} \\ |\text{UALLOW}| \text{ for upper bounds if } |\text{UALLOW}| > \text{GSCAL} \\ \text{GSCAL otherwise} \end{cases} \quad (2-11)$$

where GSCAL is an MSC Nastran parameter with a default of 0.001.

The normalized constraints of [Equation \(2-10\)](#) 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. It will be seen that this normalization enables the constraint screening discussed in [Constraint Screening](#).

Since response bounds are used as normalizing factors, it is recommended that the bounds of zero be avoided. To avoid a divide by zero error in the code, the small nonzero number GSCAL is used. This may or may not be acceptable from a design standpoint. For example, consider the synthetic response

$$\frac{\sigma_1 + \sigma_2}{2} \leq \sigma_{max} \quad (2-12)$$

and assume the response had been formulated using a type-2 response as

$$\frac{\sigma_1 + \sigma_2}{2} - \sigma_{max} \leq 0 \quad (2-13)$$

At first glance, this seems acceptable since it is expressed in standard form. However, the upper bound normalization uses GSCAL and the resultant normalized upper-bound constraint then becomes

$$g_u = \frac{\left(\frac{\sigma_1 + \sigma_2}{2} - \sigma_{max} \right)}{GSCAL} \leq 0 \quad (2-14)$$

To understand why this is not a good formulation, let's assume σ_{max} is 10000, and the $(\sigma_1 + \sigma_2)/2$ responses exceeds this by 1% to equal 10100. With the default value of GSCAL, [Equation \(2-14\)](#) gives a constraint value of 100,000(!), implying a grossly violated constraint. Further, a small change in the design could result in $(\sigma_1 + \sigma_2)/2 = 9900$ and now the constraint is -100,000. The optimizer will have a difficult time making sense out of these wild gyrations and is likely to become confused.

If the constraint is expressed instead as

$$\frac{\sigma_1 + \sigma_2}{2} \leq \sigma_{max} \quad (2-15)$$

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



$$g_u = \frac{\left(\frac{\sigma_1 + \sigma_2}{2}\right) - \sigma_{max}}{|\sigma_{max}|} \leq 0 \quad (2-16)$$

now a 1% excedence by $(\sigma_1 + \sigma_2)/2$ of σ_{max} results in a constraint value of 0.01 and the optimizer will have a much better chance of doing its job.

Equality Constraints in MSC Nastran

Equality constraints can be defined simply by specifying equivalent upper and lower bounds. This will yield a pair of equal and opposite constraints that force the response to be satisfied with equality at the optimum

$$\begin{aligned} R_j &\leq r_j(\mathbf{X}) \leq R_j \\ g_{2j-1}(\mathbf{X}) &= \frac{R_j - r_j(\mathbf{X})}{|R_j|} \leq 0 \\ g_{2j}(\mathbf{X}) &= \frac{r_j(\mathbf{X}) - R_j}{|R_j|} \leq 0 \end{aligned} \quad (2-17)$$

In practice, this is not a good idea since the optimizer does not have any special algorithms for dealing with equality constraints so that one of the two relations in [Equation \(2-17\)](#) will always be violated. It is recommended that the equality constraint be replaced with a tolerance by expressing the bounds as $R_j - \delta$ and $R_j + \delta$ on the DCONSTR entry, where δ is the allowable tolerance and is perhaps 1% of the target value.



Multidisciplinary Analysis

A structural design requires the synthesis of design requirements that can arise from any number of areas, including strength, stiffness, and stability. Historically, this often led to additional design cycles as additional analyses revealed shortcomings in the designs that were based on preliminary analyses with a limited set of requirements. As an example from aircraft design, the term “flutter penalty” was applied to the extra weight that was required in the design to satisfy flutter speed requirements once the initial strength design had been performed. The MSC Nastran implementation of design sensitivity and optimization recognizes this by combining analyses that were previously considered separate (with separate input files and job submittals) into a single job. The responses from these combined analyses can then be included in the optimization step so that all the applicable requirements are considered simultaneously.

Table 2-5 indicates the analyses that are available for design sensitivity and optimization and lists some of the attributes of the analyses. The column of Solution Numbers refers to the number MSC Nastran would use to perform the separate analyses. The Design Optimization Solution Number is 200.

Table 2-5 Analysis Disciplines Supported in Multidisciplinary Analysis and Design of SOL 200

Analysis	SOL Numbers	Multiple Subcases	Multiple Boundary Conditions
Statics	101	Y	Y
Normal Modes	103	Y	Y
Buckling	105	Y	Y
Direct Complex Eigenanalysis	107	N	N
Direct Frequency	108	Y	Y
Modal Complex Eigenanalysis	110	N	N
Modal Frequency	111	Y	Y
Modal Transient	112	N	N
Static Aeroelasticity	144	Y	Y
Flutter	145	Y	Y

It is seen that SOL 200 has value simply in the performance of multidisciplinary analysis. In fact, clients have been known to use SOL 200 to submit combined analyses without any design sensitivity or optimization simply because it simplifies the maintenance of input data and provides some performance efficiency (that is, operations are not repeated for the separate analyses). The capabilities (and limitations) of the separate Solution Numbers are incorporated into SOL 200 for the most part. **Table 2-5** indicates that many of the analysis types support multiple subcases and/or multiple boundary conditions. Analysis type Buckling requires a Static subcase. Analyses from SOLs 103, 110, 111, 112 and 145 require a normal modes analysis. When a SOL 200 job with multiple subcases is submitted, a determination is



made of the number of unique eigenanalyses that are required based on the boundary conditions and the eigenanalysis method selected in the separate subcases. Only the unique analyses are performed with the results shared with the remaining subcases.

Response Calculation

Type-1 Responses

The type-1 responses are, in most cases, response quantities that are available as part of standard MSC Nastran results processing such as displacements, stresses and eigenvalues. These responses are recovered from standard MSC Nastran tables. Several of the responses are not standard MSC Nastran results and their development requires additional description.

Weight Response

This response is calculated using the grid point weight generator that is used to produce the weight and balance information that is printed based on PARAM,GRDPNT. The WEIGHT type-1 response is identified by the selection of the row and column of the 6X6 rigid body mass matrix that is to be used for the response.

Volume Response

This response is calculated by summing the individual volumes of each of the finite elements.

WMPID Response

This response provides the weight of a designated material ID or a material ID/property ID combination.

PSD Response

PSD responses are included as available design responses and the development of these responses uses the same theoretical development as that given in the *MSC Nastran Dynamic Analysis User's Guide*. EQ# (8-10) of that document, the power spectral density of the an output response quantity is

$$S_j(\omega) = \sum_a \sum_b H_{ja}(\omega) \cdot H_{jb}^*(\omega) \cdot S_{ab} \quad (2-18)$$

where a and b denote two loading conditions (two subcases) and the superscript * indicates the complex conjugate of the response.

As an example, with two subcases, $a = 1$ and $b = 2$, the spectral density is

$$\begin{aligned} S_j(\omega) = S_{jf} = & S_{11}H_{j1}H_{j1}^* + S_{12}H_{j1}H_{j2}^* \\ & + S_{21}H_{j2}H_{j1}^* + S_{22}H_{j2}H_{j2}^* \end{aligned} \quad (2-19)$$

Note: Because the term $S_{21} = S_{12}^*$, the resulting spectral density, $S_j(\omega)$ is a real number. Individual terms of $S_j(\omega)$ can be specified as PSD responses.



RMS Response

RMS responses are the RMS value, \bar{u}_j , for the response quantity, j . This is theoretically developed as an integral that is implemented in MSC Nastran using a numerical approximation

$$\bar{u}_j = \sqrt{\int_f |S_{jf}| df} \approx \sqrt{\sum_f C_j S_{jf}} \quad (2-20)$$

where, from the trapezoidal integration method, the frequency-dependent coefficients, S_{jf} , ($f = f_i$) are multiplied by the frequency factors

$$\begin{aligned} C_{fi} &= \frac{1}{2}(f_{i+1} - f_{i-1}) & 1 < i < N \\ C_{f1} &= \frac{1}{2}(f_2 - f_1) & i = 1 \\ C_{fN} &= \frac{1}{2}(f_N - f_{N-1}) & i = N \end{aligned} \quad (2-21)$$

Type-2 Responses

The type-1 responses of the previous paragraphs provide responses that are directly available from MSC Nastran. Type-2 (or synthetic) responses, on the other hand, provide responses that are not directly available from MSC Nastran but can be formulated using a combination of arguments and user-defined equations. A number of predetermined functions are available for this response type. Please see the FUNC character strings table in chapter 4 on [page 168](#) for more information. These eliminate the task of preparing user-defined equations. [DRESP2 Bulk Data Entries](#) provides details on constructing these responses.

Type-3 Responses

This type of response has been provided for situations where the desired design response is beyond what is available using the type-2 response. The type-3 (or external) response applies the same client-server technology that has been used in MSC Nastran for user-supplied beam libraries and for p-element geometry. This enables, for example, the incorporation of proprietary local buckling criteria that are too complex to be developed using a DRESP2. [DRESP3 Bulk Data Entries](#) provides detail on constructing these responses.

Added Benefit of Synthetic and External Responses

SOL 200 can offer a benefit in the analysis phase that is in addition to the features available in the separate solution numbers. This benefit relates to [The Design Model](#) and, in particular, the type-2 and type-3 responses. The results of evaluation of the responses can be reported in the output file and the user could exploit this by developing responses that are in addition to what are available from standard data recovery. As an example, suppose it is desired to assess the buckling behavior of a column that is contained within a large finite element model. One could develop a type-2 response that is some combination of element response, element dimensions, materials, geometry, and restraint conditions.



Normally, this type of calculation would require some type of postprocessing operation on the part of the user, but this could be automated within SOL 200 to provide this result as part of the standard analysis.

Mode Tracking

The selection of a particular eigenvalue from a normal modes analysis as a design response is done by specifying its mode number. The user's intent when specifying this number may be to design this particular mode number regardless of the physical nature of the original mode shape. However, in many applications, most notably automotive, the intent is to control a particular physical mode; for example, the vibration of the vehicle's roof. In this case, the user's intent is actually to design the frequency of the physical mode shape and the mode number could vary due to the redesign. (That is, the example roof mode may be the 10th mode of the initial design, but after a redesign it becomes the 12th.)

The eigenvalue analysis of SOL 200, includes a feature to "track" the modes so that the modes are ordered based on their original physical behavior rather than on their frequency order. This is done by constructing cross-orthogonality check between the current design cycle and the previous cycle

$$\Phi'^T_i M_i \Phi'_{i-1} = t_i \quad (2-22)$$

where the subscript i denotes the current design cycle, $i - 1$ the previous, and the prime over Φ indicates mass-normalized eigenvectors.

If there had been no changes in either mode shape, order, or mass of the system, the cross-orthogonality check would yield a diagonal matrix. If the mode shapes change due to a mass and/or stiffness variation, however, their correlation will be less than 1.0. If the mode numbers have changed, dominant off-diagonal terms will be present in t_i . Where sufficient correlation exists, the results of this cross-orthogonality check can be used as a basis to determine which modes have switched.



Constraint Screening

The concept of constraint screening was introduced in a general way in [Structural Optimization](#). The idea is to limit the constraints that are considered in the redesign to those that are likely to play an active role in the redesign. To motivate the importance of this, consider an example where the user has defined four stress responses (say σ_x , σ_y , τ_{xy} and von Mises) at the top and bottom of each QUAD4 finite element and say the model contains 10000 of these elements. Further, suppose 10 static loads cases represent the design loads for the design task. This rather modest problem statement results in 4 stresses * 2 surfaces * 10000 elements * 10 subcases * 2 constraints (upper and lower bounds) = 1.6 million constraints. Clearly, it is not practical or desirable to consider all of these constraint conditions when performing a design task; hence the use of constraint screening.

[The Design Model](#) discussed how the normalization of the design constraints has put all the design conditions on an equal basis regardless of whether the underlying design response is a pressure, displacement, stress or whatever. The screening process is a two stage process that does an initial screening based on a threshold and a second screening based on the region. The threshold screening simply says that it is possible to ignore constraints unless they exceed a specified value.

[Figure 2-4](#) 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.

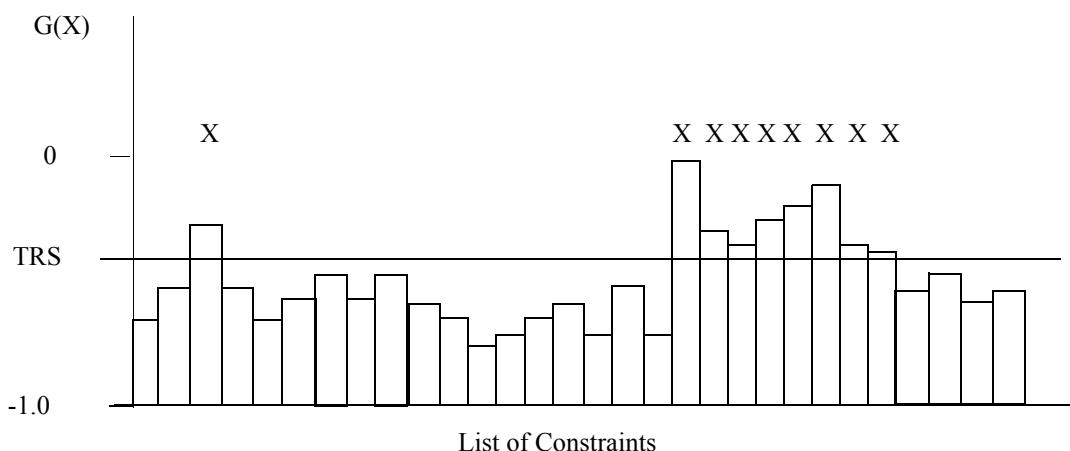


Figure 2-4 Constraint Deletion

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.

As an example, suppose we have an automobile roof panel, modeled with a number of quadrilateral elements. This panel 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 panel since all of these stresses will vary nearly in unison as the panel thickness is varied. Thus, it is probably safe to retain only a few of the largest valued constraints from this “region.”

Constraint regionalization is shown in [Figure 2-5](#), which is the same as [Figure 2-4](#), 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.

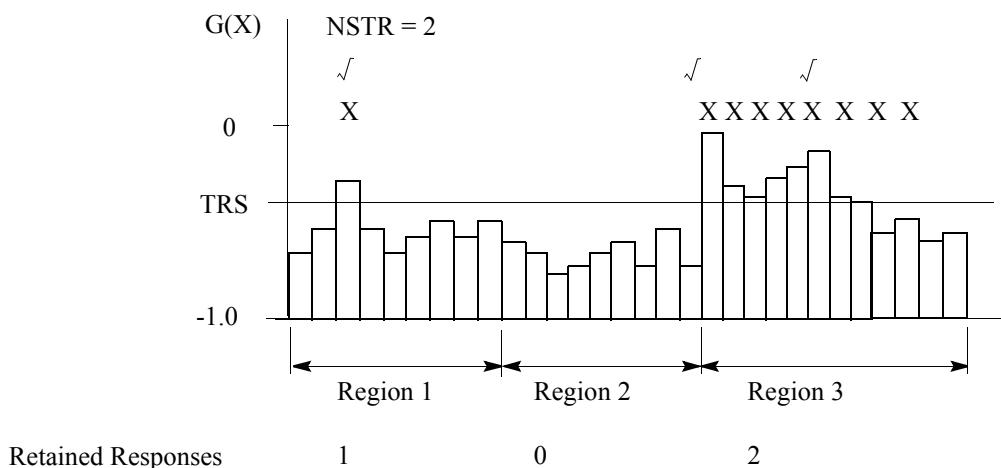


Figure 2-5 Constraint Regionalization

It should be apparent that this screening process could result in a dramatic reduction in the 1.6 million constraints mentioned in the example above; perhaps to as few as 100 retained constraints. This is a much more tractable problem for the optimization algorithm to deal with. There is another more subtle, but equally important, benefit that can be realized from constraint screening. The example given above assumed that 10 load cases were used to develop the constraints. It is quite conceivable that some of these loads are unimportant in terms of producing critical design responses. The entire load case can then be eliminated from consideration in the sensitivity phase of the analysis with a direct effect on the performance of the sensitivity task of [Design Sensitivity Analysis](#). This concept is referred to as “load case deletion” and plays an important role in MSC Nastran design sensitivity and optimization.

As a final comment on constraint screening, sometimes it is desirable to retain constraints that would not normally survive the screening process. For example, perhaps only a sensitivity analysis is being performed and it is desired to know the sensitivity of all the responses, regardless of the value of the constraints. In another case, it might happen that a particular constraint is very sensitive and while not critical for a particular design, becomes critical, or even violated after a redesign and then gets deleted



again after the second redesign. The design could then bounce back and forth between two designs (one with the constraint well satisfied but with a large objective and the second with the constraint violated but with a small objective) and not converge. For these reasons, the threshold parameter TRS and the regionalization parameter NSTR are made accessible to the user and can be adjusted to increase the likelihood that a particular response is retained.



Design Sensitivity Analysis

Design sensitivity analysis, as defined in this document, computes the rates of change of structural response quantities with respect to changes in design variables. Other sensitivities, such as change in a constraint value due to a change in the design variable or change in the response due to a change in a property, can be derived from chain rule operations on the sensitivities featured here. 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 MSC Nastran whenever design optimization is requested. There may be times when you want MSC Nastran 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 MSC Nastran. This is a broad subject that is organized by first presenting some general concepts that are applicable to all of the sensitivity analyses. This is followed by a detailed discussion of element and grid responses. Adjoint and direct techniques for sensitivity are detailed and then sensitivity analyses for statics, dynamics and static aeroelasticity are described. This is followed by a discussion of the sensitivity of a number of responses that are global quantities: weight/volume, eigenvalues (normal modes and buckling), complex eigenanalysis, and flutter.

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 $\partial r / \partial x$. These coefficients are evaluated at a particular design characterized by the vector of design variables \mathbf{x}^0 , giving

$$\lambda_{ij} = \left. \frac{\partial r_j}{\partial x_i} \right|_{\mathbf{x}^0} \quad (2-23)$$

where subscripts are used to indicate the i -th design variable and the j -th response. [Equation \(2-23\)](#) is just the slope of the response with respect to x_i as is shown in [Figure 2-6](#).



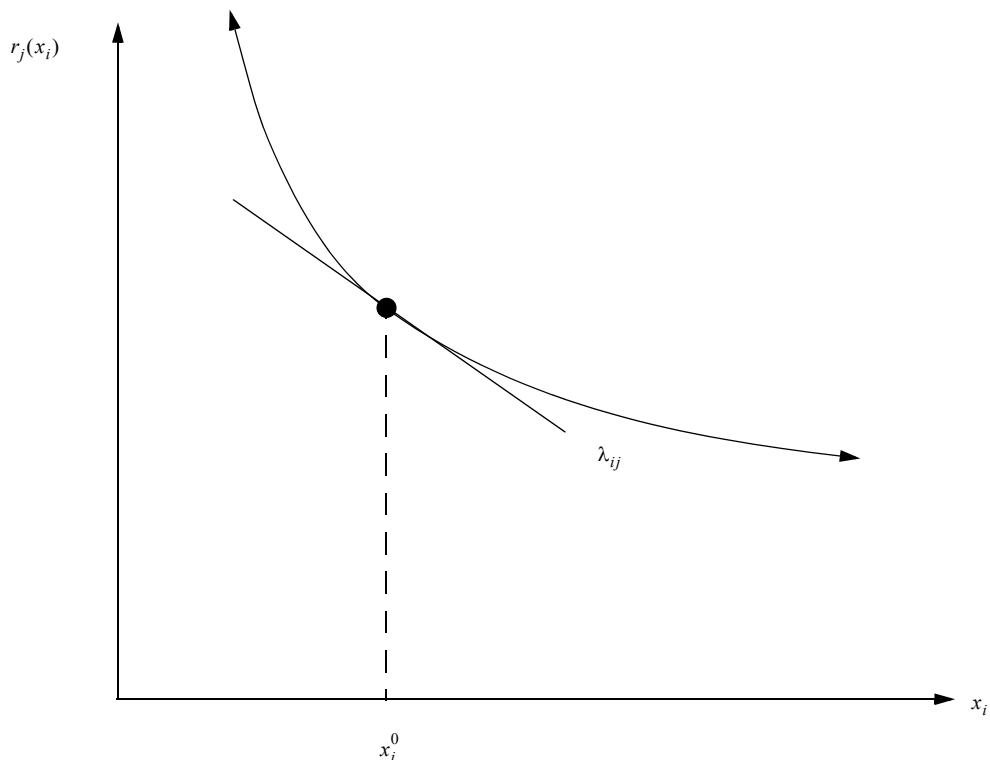


Figure 2-6 Design Sensitivity Coefficient -- Graphical Interpretation

For Which Responses Are Sensitivities Computed?

Sensitivities are computed for all of the responses (types-1, 2, and 3) that are used to define the objective function and retained constraints.

Due to the constraint screening described in [Constraint Screening](#), fewer response sensitivities are typically computed than there are responses defined in the design model.

Internal Representation of the Sensitivity Coefficients in MSC Nastran

For reasons of efficiency as well as accuracy, MSC Nastran uses a slightly different internal representation of the coefficients of [Equation \(2-23\)](#) depending on how the design variables are related to the analysis model properties.

The set of independent design variables is taken as the basis for the design sensitivity coefficients when the linear relation between the designed property and design variable is used. This is an efficient choice since a large number of properties may be a function of a much smaller set of design variables.



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

$$\lambda_{ij} = \left. \frac{\partial r_j}{\partial p_i} \right|_{\mathbf{x}^0} \quad (2-24)$$

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 known, 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 MSC Nastran are

$$\frac{\partial r_j}{\partial x_i} \quad i = 1, NDVI$$

$$\frac{\partial r_j}{\partial p_k} \quad k = 1, NIPR \quad (2-25)$$

where *NDVI* is the number of independent design variables and *NIPR* is the set of designed properties which are a nonlinear function of the design variables (including the “spawned” designed properties from the beam library).

To illustrate the advantage in using a mixed design variable-property basis, consider responses that depend on properties which are linear functions of the design variables. The approximation for a change in the response due to a change in the design variable is

$$r_j(\mathbf{X}^0 + \Delta x_1) \approx r_j(\mathbf{X}^0) + \left. \frac{\partial r_j}{\partial x_1} \right|_{\mathbf{x}^0} \cdot \Delta x_1 \quad (2-26)$$

where the notation $(\mathbf{X}^0 + \Delta x_i)$ is used to indicate that the *i*th design variable is being perturbed while the remaining variables on the \mathbf{X}^0 vector are unchanged.

For responses that depend on the properties that are nonlinear functions of the design variables, the approximation can be written as

$$r_j(\mathbf{X}^0 + \Delta x_i) = r_j(\mathbf{X}^0) + \sum_k \left. \frac{\partial r_j}{\partial p_k} \right|_{\mathbf{x}^0} \cdot (p_k(\mathbf{X}^0 + \Delta x_i) - p_k(\mathbf{X}^0)) \quad (2-27)$$

where the properties at the perturbed design in [Equation \(2-27\)](#) can be evaluated precisely from the input equations.



Finite Difference Methods

Finite difference techniques are used in places in MSC Nastran where it is not practical to compute analytical sensitivities.

MSC Nastran employs two alternative finite difference schemes in its sensitivity calculation: forward difference and central difference. Conceptually, forward difference techniques determine the slope of a function $f(\mathbf{X})$ at point \mathbf{x}^0 by solving

$$\frac{\partial f(\mathbf{X}^0)}{\partial x_i} = \frac{f(\mathbf{X}^0 + \Delta x_i) - f(\mathbf{X}^0)}{\Delta x_i} \quad (2-28)$$

while central differencing solves for

$$\frac{\partial f(\mathbf{X}^0)}{\partial x_i} = \frac{f(\mathbf{X}^0 + \Delta x_i) - f(\mathbf{X}^0 - \Delta x_i)}{2\Delta x_i} \quad (2-29)$$

The forward differencing provides an estimate of the true sensitivity with an error that is on the order of the Δx step size. Central differencing reduces the expected error to be on the order of Δx^2 . Central differencing therefore provides a higher quality result but at the added cost of two difference steps rather than one.

Sensitivities of type-2 and type-3 responses are always calculated using central differencing techniques to obtain the change in the response due to the change in one of the inputs. Chain rule techniques are employed to obtain the overall sensitivity of one of these responses to the design variable. For example, suppose one has constructed a type-2 response that is a function of a single displacement, stress and design variable

$$r_j = r_2(u_r, \sigma_s, x_t) \quad (2-30)$$

Then the sensitivity with respect to the i th design variable is computed using

$$\frac{dr_j}{dx_i} = \frac{\partial r_j}{\partial x_t} \frac{\partial x_t}{\partial x_i} + \frac{\partial r_j}{\partial u_r} \frac{\partial u_r}{\partial x_i} + \frac{\partial r_j}{\partial \sigma_s} \frac{\partial \sigma_s}{\partial x_i} \quad (2-31)$$

where the $\partial r_j / \partial x_t$, $\partial r_j / \partial u_r$ and $\partial r_j / \partial \sigma_s$ terms are calculated using central differencing techniques and $\partial u_r / \partial x_i$ and $\partial \sigma_s / \partial x_i$ terms are derived from the response sensitivity analysis described in the following sections. The $\partial x_t / \partial x_i$ term is 1.0 for $i = t$ and 0.0 otherwise.

Element and Grid Responses

Direct and Adjoint Sensitivity Methods

MSC Nastran supports two distinct methods for computing sensitivity coefficients: direct sensitivity and adjoint sensitivity. The difference between the two methods can be motivated by expressing a response as a function of the finite element responses and the design variables

$$r = f(u, \mathbf{X}) \quad (2-32)$$



The sensitivity of this response with respect to the design variables can be expressed as

$$\frac{\partial r}{\partial x} = \frac{\partial f}{\partial u} \frac{\partial u}{\partial x} \quad (2-33)$$

The direct method solves for $\partial u / \partial x$ and uses that in a semi-analytic formulation such as that given in [Direct Sensitivities of Static Responses](#) to develop the response sensitivity. If the system equation for the finite element analysis can be written in a general form as

$$[S]\{u\} = \{P\} \quad (2-34)$$

where S is simply K in a statics analysis and is $-\omega^2 M + i\omega B + K$ in a frequency response analysis, then the derivative of [Equation \(2-34\)](#) gives

$$[S] \left\{ \frac{\partial u}{\partial x} \right\} = \left\{ \frac{\partial P}{\partial x} \right\} - \left[\frac{\partial S}{\partial x} \right] \{u\} \quad (2-35)$$

The important point about [Equation \(2-35\)](#) for now is that it requires the solution of $ndv \cdot nlc$ right-hand sides, where $NDV = NDVI + NIPR$ of [Equation \(2-25\)](#) and nlc is the number of retained $\{u\}$ solution vectors. For statics solutions, this is the number of retained subcases while for a frequency response analysis, it is the number of retained frequencies per subcase summed over the number of subcases.

The adjoint method employs a scheme that first solves for adjoint solutions using

$$[S]^T \{\lambda\} = \left\{ \frac{\partial f}{\partial u} \right\} \quad (2-36)$$

and then substitutes in [Equation \(2-33\)](#) to give

$$\frac{\partial r}{\partial x} = \{\lambda\}^T \left\{ \frac{\partial P}{\partial x} - \left[\frac{\partial S}{\partial x} \right] \{u\} \right\} \quad (2-37)$$

Where $[S]^{-T}[S] = [I]$ has been used to eliminate the system matrix.

[Equation \(2-36\)](#) requires the solution of $nresp$ equations, where $nresp$ is the number of retained responses. There is an important class of problems where the adjoint method has a distinct advantage over the direct method: frequency response optimization. As an example, consider a design model that has 20 design variables and 100 frequency excitation points that results in 150 active responses distributed over the 100 frequencies. (Note that excitation frequencies that do not result in active responses are screened out and do not contribute to the nlc calculation.) In this case, the adjoint method requires 150 solutions while the direct method requires 2000. This indicates that the adjoint method can be expected to require roughly 7.5% of the resources (CPU and disk space) needed by the direct method.

This section continues by discussing the direct sensitivity method with a subsequent discussion of the adjoint method. The adjoint method has been implemented in MSC Nastran for grid responses.



Direct Sensitivities of Static Responses

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 the displacement is written as,

$$r = r(\mathbf{U}, \mathbf{X}) \quad (2-38)$$

For the direct method in MSC Nastran, the sensitivities of these responses with respect to changes in the design variables are approximated using first forward finite differences as

$$\frac{\partial r_j}{\partial x_i} \equiv \frac{\Delta r_j}{\Delta x_i} = \frac{r_j(\mathbf{X}^0 + \Delta x_i, \mathbf{U} + \Delta \mathbf{U}) - r_j(\mathbf{X}^0, \mathbf{U}^0)}{\Delta x_i} \quad (2-39)$$

where r_j is the j -th design response. It should be noted that [Equation \(2-39\)](#) uses \mathbf{x} and Δx in a generic way to encompass the non-linear properties of [Equation \(2-25\)](#). The steps required to evaluate [Equation \(2-4\)](#) include:

1. The designed properties and/or designed shape basis vectors are perturbed using a finite difference step and perturbed element matrices are developed.
2. Pseudo-loads are constructed from the perturbed loads, the perturbed element matrices and the solution vectors.
3. The perturbed displacements resulting are solved from the pseudo-loads.
4. The perturbed displacements are added to the baseline solution vectors to produce the total $\mathbf{U} + \Delta \mathbf{U}$ solution vectors.
5. Data recovery is applied to these total vectors to obtain the responses when the design quantities have been perturbed.
6. [Equation \(2-39\)](#) is applied to obtain the required sensitivity.

Each of these steps is now described in greater detail.

Property and/or Shape Basis Vector Perturbation

For property optimization, the size of the move Δx is given by

$$\Delta x_i = \text{DELB} \cdot x_i \quad (2-40)$$

or

$$\Delta p_i = \text{DELB} \cdot p_i \quad (2-41)$$

where DELB is user accessible parameter with a default value of 0.0001. If x_i in [Equation \(2-40\)](#) or p_i in [Equation \(2-41\)](#) is exactly zero, then the respective move is given by .001.

For shape basis vectors, the selection of the step size is more involved because, as the discussion on designed shapes in [The Design Model](#) indicates ([Equation \(2-11\)](#)), the magnitude of the change in the grid location is the product of the shape basis vector and the change in the design variable.



For shape sensitivities, grid coordinates are perturbed a finite amount

$$\{\delta G\}_i = \frac{STPSCL}{E_i} \{T\}_i \quad (2-42)$$

The subscript i in [Equation \(2-42\)](#) 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 user accessible parameter with a default value of 1.0.

The maximum strain energy norm used in [Equation \(2-42\)](#) is computed by first determining the grid “forces” due to a shape basis vector “displacement”

$$[K]\{T\}_i = \{F_s\} \quad (2-43)$$

Note that the components of T associated with rotational DOFs (degrees-of-freedom) are set to zero so that no forces are generated on these DOFs. The forces can be used to compute a strain energy per grid location as

$$\{F_s\}\{T\}_i = \{E\}_i \quad (2-44)$$

[Equation \(2-44\)](#) is an open product, resulting in a vector of strain energies. Locating the maximum term in the vector and taking its square root yields

$$E_i = (\max\{E\}_i)^{1/2} \quad (2-45)$$

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 sensitivity.

Pseudo-Load Vectors

The term Pseudo-Load vectors is used to refer to the terms on the right hand side of [Equation \(2-35\)](#). For static analysis this equation, can be written as

$$[K] \frac{\partial\{U\}}{\partial x_i} = \frac{\partial\{P\}}{\partial x_i} - \frac{\partial[K]}{\partial x_i} \{U\} \quad (2-46)$$

Computationally, [Equation \(2-46\)](#) and [Equation \(2-34\)](#) are identical except that they have different right-hand side load vectors, hence the term pseudo-loads. 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 reused. This observation is taken advantage of in MSC Nastran.

Note that the right-hand side of [Equation \(2-46\)](#) 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.

The selection of the finite difference method (forward vs. central) is controlled in MSC Nastran by PARAM CDIF. The default is ‘NO’ (forward differencing) for property sensitivity and ‘YES’ (central



differencing) for shape sensitivity. These defaults reflect MSC's experience that it is generally more difficult to obtain accurate shape sensitivities relative to the property sensitivities.

- For forward differencing,

$$\begin{aligned}\frac{\partial[K]}{\partial x_i} \{u\} &\equiv \frac{[K(\mathbf{X}^0 + \Delta x_i)] - [K(\mathbf{X}^0)]}{\Delta x_i} \{u\} \\ \frac{\partial\{P\}}{\partial x_i} &\equiv \frac{\{P(\mathbf{X}^0 + \Delta x_i)\} - \{P(\mathbf{X}^0)\}}{\Delta x_i}\end{aligned}\quad (2-47)$$

- For central differencing,

$$\begin{aligned}\frac{\partial[K]}{\partial x_i} \{u\} &\equiv \frac{[K(\mathbf{X}^0 + \Delta x_i)] - [K(\mathbf{X}^0 - \Delta x_i)]}{2\Delta x_i} \{u\} \\ \frac{\partial\{P\}}{\partial x_i} &\equiv \frac{\{P(\mathbf{X}^0 + \Delta x_i)\} - \{P(\mathbf{X}^0 - \Delta x_i)\}}{2\Delta x_i}\end{aligned}\quad (2-48)$$

The product of the change in the matrices times the solution vectors is formed directly rather than first forming the difference in the matrices and performing the multiplication by the solution vectors later. This greatly improves the performance and disk space usage in the sensitivity calculation.

In most cases, the load vector is invariant with respect to design variable changes. An exception is if the loads are being explicitly designed using the parameters available in Table 2-4. Additionally, gravity loads can vary if the mass of the structure is changed, while thermal loads can also change as a function of element properties. MSC Nastran does not account for changes in loading due to changes in shape in its sensitivity calculations so that, for example, a pressure load is assumed to be invariant in the shape sensitivity calculation. The changed load is accounted for when a finite analysis is performed on the redesigned shape.

Solution for the Perturbed Displacements

Once the pseudo-loads have been developed, standard MSC Nastran solution techniques can be used to solve for the perturbed Δu displacements. This entails reduction techniques to bring the loads to the degrees-of-freedom required by the decomposed stiffness matrix and subsequent recovery to bring the Δu vectors back to the full set of physical degrees-of-freedom.

Addition of Perturbed and Baseline Solution Vectors

It needs to be recognized that in general

$$r(u + \Delta u) \neq r(u) + r(\Delta u) \quad (2-49)$$

so that it is necessary to form the $\{u + \Delta u\}$ for the recovery of the perturbed responses. This entails a straightforward addition except that the $\{\Delta u\}$ vector has columns for each design variable for each column in $\{u\}$. Therefore it is necessary to precede the addition by an operation which duplicates the $\{u\}$ vectors to make this matrix conformable with the $\{\Delta u\}$ vectors.



Data Recovery

Standard MSC Nastran data recovery techniques are applied to extract retained design responses for each of the perturbed design variables.

Final Sensitivity Calculations

With the perturbed and baseline responses, it is now possible to apply [Equation \(2-39\)](#) to compute the actual sensitivity of the type-1 one responses with respect to the independent design variables and the nonlinear properties.

Inertia Relief Sensitivity Analysis

A special case in static sensitivity analysis occurs when the structure being analyzed is free to move as a rigid body. In this case “inertia relief” techniques are applied, resulting in additional terms in the sensitivity calculations. A complete derivation of the sensitivity calculations for this case is not warranted; however, a sketch of the method, indicating how inertia relief sensitivity analysis is implemented in MSC Nastran, is provided.

For static inertia relief analysis, a revised loading is developed using

$$P^I = P^L - MD\ddot{u}_r \quad (2-50)$$

where

L = the applied loading.

D = a “rigid body mode” matrix that provides accelerations at all grid points due to rigid body accelerations \ddot{u}_r .

These accelerations can be either specified by the user with a DMIG,UACCEL Bulk Data entry or computed as part of the solution using

$$\{\ddot{u}_r\} = [M_r]^{-1}\{P_r^I\} \quad (2-51)$$

where the mass and loading terms have been reduced to the “SUPPORT”ed degrees-of-freedom.

The sensitivity of the load is then

$$\{\Delta P^I\} = \{\Delta P^L - \Delta MD\ddot{u}_r - MD\Delta\ddot{u}_r - \Delta MD\Delta u_r\} \quad (2-52)$$

D is considered invariant in this derivation, indicating that shape changes are not supported. The Δu_r term is calculated from a perturbation of [Equation \(2-51\)](#) to give

$$\{\Delta u_r\} = [M_r + \Delta M_r]^{-1}\{\Delta P_r^L - \Delta M_r\ddot{u}_r\} \quad (2-53)$$

Note that second order terms, such as $\Delta MD \Delta u_r$, are retained in this special analysis.



Shape Sensitivity with Rigid Elements

Rigid elements provide a convenient way of specifying linear relationships among degrees-of-freedom. This has implications in shape sensitivity analysis since the perturbations in the grid locations produce changes in the linear relationships. Basically, the relationship between dependent (m-set) and independent (n-set) degrees-of-freedom is expressed as

$$[R_m]\{u_m\} + [R_n]\{u_n\} = 0 \quad (2-54)$$

This gives rise to a relationship between dependent and independent degrees-of-freedom of the form

$$\{u_m\} = [G_m]\{u_n\} \quad (2-55)$$

where

$$[G_m] = -[R_m]^{-1}[R_n] \quad (2-56)$$

or

$$[R_m][G_m] = -[R_n] \quad (2-57)$$

For sensitivity analysis, it is then necessary to develop $[\Delta G_m]$ from perturbing [Equation \(2-57\)](#)

$$[\Delta R_m][G_m] + [R_m][\Delta G_m] = -[\Delta R_n] \quad (2-58)$$

or

$$[\Delta G_m] = -[R_m]^{-1}([\Delta R_n] + [\Delta R_m][G_m]) \quad (2-59)$$

If $[R_n]^+$ and $[R_m]^+$ are defined as being finite forward steps for the R_n and R_m matrices, then [Equation \(2-57\)](#) can be used in [Equation \(2-59\)](#) to give

$$[\Delta G_m] = -[R_m]^{-1}([R_n]^+ + [R_m]^+[G_m]) \quad (2-60)$$

The rigid element sensitivity calculation, therefore, entails including this $[\Delta G_m]$ when computing the pseudo loads. The details of including these terms are not shown here, but differ by analysis type and whether direct or adjoint methods are employed.

Static Aeroelastic Sensitivity Analysis

Static Aeroelastic analysis performs a static analysis of a free-flying vehicle immersed in an airflow. This is a separate discipline within MSC Nastran SOL 200 and the available responses are:

- Static aeroelastic responses, including not only displacement, stresses, forces, etc., but also trim variables. This allows the evaluation, for example, the change in the trim angle of attack due to a structural change.
- 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.



- In addition to support for sizing, shape and topology optimization, Static Aeroelasticity supports the design of control surfaces settings.

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 *MSC Nastran Aeroelastic Analysis User's Guide*.

The form of the equations is the same as for sensitivity analysis of static responses

$$[K]\{u\} = \{P\} \quad (2-61)$$

where, for static aeroelastic analysis,

$$\begin{aligned} [K] &= \begin{bmatrix} K_{ll} + K_{ll}^a & K_{lr} + K_{lr}^a & M_{ll}D + M_{lr} & K_{lx}^a \\ D^T M_{ll} + M_{rl} & D^T M_{lr} + M_{rr} & 0 & 0 \\ D^T K_{ll}^a + K_{rl}^a & D^T K_{lr}^a + K_{rr}^a & m_r & D^T K_{lx}^a + K_{rx}^a \\ 0 & 0 & 0 & 0 \end{bmatrix} \\ [u] &= \begin{Bmatrix} u_l \\ u_n \\ \ddot{u}_r \\ u_x \end{Bmatrix} \\ \{P\} &= \begin{Bmatrix} P_l \\ 0 \\ P_r + D^T P_l \\ P_x \end{Bmatrix} \end{aligned} \quad (2-62)$$

This equation is differentiated with respect to a design variable and solved for the “displacement” sensitivities using a semi analytic approach as has been described in [Equation \(2-46\)](#) for static sensitivity analysis. Static Aeroelastic Sensitivity (see [Aeroelastic Design Sensitivities and Optimization](#) in Chapter 2 of the *MSC Nastran Aeroelastic Analysis User's Guide*) provides further details of the sensitivity and indicates the special steps that are required to accommodate the mean-axis constraint embedded in [Equation \(2-62\)](#). Also, the stability derivative responses necessitate the generation of additional solution vectors for a unit deflection of the associated aerodynamic variable.

Direct Sensitivities for Dynamics

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



All of these disciplines support fluid-structural coupling.

General Considerations

The general methodology employed for dynamic responses are very similar to those just described for static analysis so that this description uses far less detail and the reader is referred to the previous section for a more complete description.

The complete stiffness matrix for dynamic response analysis consists of a superposition from damping as well as direct matrix input

$$[K] = [K_1] + ig[K_1] + i \sum g_e [K_e] + [K_2] \quad (2-63)$$

$[K_1]$ = structural stiffness

g = uniform structural damping coefficient PARAM,G

g_e = structural damping coefficient, GE, on a MATi entry

$[K_2]$ = direct matrix input at grids DMIG

The DMIG input of K_2 cannot be designed, nor can PARAM G. The K_1 matrix can be affected by any of the available property or shape design parameters. The g_e parameter is accessible to design as one of the material properties shown in [Table 2-2](#).

The total damping matrix is the sum of the contributions from viscous elements $[B_1]$ and direct matrix input at the grids $[B_2]$

$$[B] = [B_1] + [B_2] \quad (2-64)$$

For structural mass,

$$[M] = [M_1] + [M_2] \quad (2-65)$$

The direct matrix inputs for damping $[B_2]$ and mass $[M_2]$ are assumed constant in the sensitivity analysis as is the case with the direct matrix input for stiffness.

Direct Frequency Response Sensitivity Analysis

Sensitivities for direct frequency response are obtained by differentiating the governing equation

$$(-\omega^2[M] + i\omega[B] + [K])\{U\} = \{P\} \quad (2-66)$$

to obtain

$$(-\omega^2[M] + i\omega[B] + [K])\{\Delta U\} = \{\Delta P\} - [-\omega^2[\Delta M] + i\omega[\Delta B] + [\Delta K]]\{U\} \quad (2-67)$$

where Δ indicates differentiation with respect to a design variable, or $\partial/\partial x_i$. [Equation \(2-67\)](#) is solved once for each forcing frequency, load condition, and design variable.



Although the force derivative term $\{\Delta P\}$ is shown in [Equation \(2-67\)](#) for completeness, this term is assumed to be zero in MSC Nastran. Therefore, if gravity, thermal, or geometry dependent loadings are significant, some error will be introduced into the approximation. This assumption of zero $\{\Delta P\}$ also applies to the modal frequency and modal transient formulations.

There is a subtle issue in the data recovery step of the dynamic sensitivity calculation that can be mentioned here. MSC Nastran calculates results in the analysis set and then recovers them to the physical set. One step in this standard recovery is the calculation of the omitted degrees-of-freedom using a relation of the form

$$\{u_o\} = [G_o]\{u_a\} \quad (2-68)$$

(See [Equation \(18-33\)](#) in the .

This ignores terms that arise from the structure's mass and the loads applied to the omitted degrees-of-freedom that is given in [Eq. \(18-30\)](#) in the .

$$\{u_o^o\} = [K_{oo}]^{-1}[P_o - [M_{oa}]\{u_a\}] \quad (2-69)$$

In a standard dynamic analysis, user's are cautioned to not apply loads to omitted degrees-of-freedom and not to omit degrees-of-freedom that have appreciable mass so that the approximation involved in [Equation \(2-69\)](#) can be considered negligible. The omitted degrees-of-freedom are typically nodal rotations that have no load or mass associated with them. However, in a sensitivity analysis, $[\Delta K]\{u\}$ terms can easily produce loads on these omitted degrees-of-freedom with the result that the sensitivities of element responses can be in significant error. MSC has provided a remedy in the form of PARAM SENSUOO. The default for this parameter is NO so that the terms of [Equation \(2-68\)](#) are neglected. Setting SENSUOO to YES includes the term. With the de-emphasis on omitted degrees-of-freedom in present day finite element analysis, this is typically not a major issue.

Modal Frequency Response Analysis

The governing equations for modal frequency response analysis are obtained from the equations for frequency response by introducing the modal transformation

$$\{U\} \cong [\Phi]\{\xi\} \quad (2-70)$$

where $[\phi]$ is a matrix of eigenvectors, perhaps augmented by residual vectors and $\{\xi\}$ is a vector of modal coordinates.

If [Equation \(2-70\)](#) is applied to [Equation \(2-66\)](#) and the result is remultiplied by $[\phi]^T$, the following set of equations result

$$(-\omega^2[\Phi]^T[M][\Phi] + i\omega[\Phi]^T[B][\Phi] + [\Phi]^T[K][\Phi])\{\xi\} = [\Phi]^T\{P\} \quad (2-71)$$

MSC Nastran also supports a modal damping term that can be added to the diagonal elements of the modal damping matrix. These terms are expressed as

$$b_i = g_i\omega_i m_i \quad (2-72)$$



where ω_i is the undamped vibration frequency, m is the generalized mass in the i th mode and g_i is a user input value. (See (6-19) in the in the [MSC Nastran Dynamic Analysis User's Guide](#). It is recommended that this form of damping be avoided in design sensitivity and optimization. This is because MSC Nastran neglects the sensitivity of this term when performing a sensitivity analysis in a modal frequency response analysis. It has been found that this can result a substantial error in the sensitivity results, particularly when the excitation frequency is near a resonant frequency. It is recommended that the PARAM G form of structural damping be used in place of the modal damping when performing modal frequency response sensitivity analysis.

An important assumption that is made is that the normal modes technique used in performing the modal analysis is equally adequate in performing a sensitivity analysis. That is, the basis that is used to span the analysis space that is used to justify a modal analysis in the first place should be just as valid in a sensitivity analysis. The benefit of making this assumption is that it avoids the need for computing the sensitivities of the eigenvectors. Support for this assumption can be obtained from the realization that modal analyses can be performed without any loss of accuracy relative to a direct analysis if a complete set of modes (that is all the modes are extracted from the physical model) is used. Therefore a sensitivity analysis that uses a complete set of modes applied to the pseudo-loads will result in the same sensitivity results as would be obtained using a direct analysis. Another way of stating this is that it assumes that

$$\{\Delta U\} \cong [\Phi]\{\Delta \xi\} \quad (2-73)$$

Applying this to [Equation \(2-67\)](#) yields

$$(-\omega^2[\Phi]^T[M][\Phi] + i\omega[\Phi]^T[B][\Phi] + [\Phi]^T[K][\Phi])\{\Delta \xi\} = \\ [\Phi]^T(\{\Delta P\} - (-\omega^2[\Delta M] + i\omega[\Delta B] + [\Delta K])\{U\}) \quad (2-74)$$

With the derivatives of the modal coordinates computed from [Equation \(2-74\)](#), the displacement derivatives are available by direct substitution into [Equation \(2-73\)](#). The data recovery steps of the static sensitivity analysis can then be applied to obtain the sensitivities of the grid and element responses.

The above discussion of the assumption that accurate sensitivities can be obtained without including eigenvector sensitivities should be qualified by pointing out that the pseudo-loads generated in a sensitivity analysis are likely to excite the structure in ways that typical structural loads do not (for example, the pseudo-loads are likely to be in the plane of the structure while applied loads are typically transverse to the structure). For this reason, it is recommended that more modes be retained when performing a sensitivity analysis than would be used in frequency response analysis. A rule of thumb is that twice as many modes should be retained to obtain accurate sensitivities as are required to obtain accurate responses. However, this is only a guideline and it is strongly recommended that a convergence study be performed by increasing the number of retained normal modes until the sensitivity values do not change significantly. It is noted in this regard that residual vectors are computed by default to degrees of freedom that are designated as design responses in a modal frequency analysis.

Modal Transient Response Sensitivities

The governing differential equation for transient dynamic response is

$$[M]\{\ddot{U}\} + [B]\{\dot{U}\} + [K]\{U\} = \{P\} \quad (2-75)$$



The modal transformation is again given by

$$\{U\} = [\Phi]\{\xi\} \quad (2-76)$$

Introducing the modal transformation into [Equation \(2-75\)](#) and differentiating with respect to a design variable yields the expression for the modal transient displacement sensitivities

$$\begin{aligned} [\Phi]^T [M][\Phi]\{\Delta\xi\} + [\Phi]^T [B][\Phi]\{\Delta\dot{\xi}\} + [\Phi]^T [K][\Phi]\{\Delta\xi\} \\ = [\Phi]^T (\{\Delta P\} - ([\Delta M]\{\ddot{U}\} + [\Delta B]\{\dot{U}\} + [\Delta K]\{U\})) \end{aligned} \quad (2-77)$$

As in modal frequency sensitivity analysis, [Equation \(2-77\)](#) makes the assumption that a sufficient number of modes are included such that the relation of [Equation \(2-73\)](#) is a good approximation.

Once the solution for the modal coordinates and their time derivatives are available, the displacement derivatives are computed from [Equation \(2-76\)](#).

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. MSC 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 [Coupled Fluid-Structure Interaction](#) (p. 752) in the *MSC Nastran Reference Manual* 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

$$[M]\{\ddot{x}\} + [K]\{x\} = \{F\} \quad (2-78)$$

The solution vector $\{x\}$ consists of both structural displacements $\{U\}$ and the fluid pressure $\{P\}$.

$$\{x\} = \begin{Bmatrix} \{U\} \\ \{P\} \end{Bmatrix} \quad (2-79)$$

The vector of external loads $\{F\}$ is comprised of both structural as well as fluid forces or

$$\{F\} = \begin{Bmatrix} \{F_s\} \\ \{F_f\} \end{Bmatrix} \quad (2-80)$$

The mass matrix in [Equation \(2-78\)](#) is defined as

$$[M] = \begin{bmatrix} [M_s] & 0 \\ -[A]^T [M_f] \end{bmatrix} \quad (2-81)$$



- $[M_s]$ = structural mass
- $[M_f]$ = fluid mass (including compressibility effects)
- $[A]$ = fluid-structure coupling matrix

The stiffness matrix consists of

$$[K] = \begin{bmatrix} [K_s] & [A] \\ 0 & [K_f] \end{bmatrix} \quad (2-82)$$

- $[K_s]$ = structural stiffness
- $[K_f]$ = effective fluid stiffness

If the loads in [Equation \(2-78\)](#) 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 [Equation \(2-67\)](#), 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

$$\begin{aligned} [u] &= [\Phi_s]\{\xi_s\} \\ [m_s] &= [\Phi_s]^T [M] [\Phi_s] \\ [k_s] &= [\Phi_s]^T [K] [\Phi_s] \end{aligned} \quad (2-83)$$

- $[\Phi_s]$ = are 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

$$\begin{aligned} \{P\} &= [\Phi_f]\{\xi_f\} \\ [m_f] &= [\Phi_f]^T [M] [\Phi_f] \\ [k_f] &= [\Phi_f]^T [K] [\Phi_f] \end{aligned} \quad (2-84)$$

where $[\Phi_f]$ are the modes of the fluid enclosed by a rigid container.

After modal reduction for both structural and fluid degrees-of-freedom, the equations of motion can be written as



$$\begin{bmatrix} m_s & 0 \\ -\Phi_f^T A^T \Phi_s & m_f \end{bmatrix} \begin{Bmatrix} \ddot{\xi}_s \\ \ddot{\xi}_f \end{Bmatrix} + \begin{bmatrix} k_s \left(\Phi_s^T A \Phi_f \right) \\ 0 \end{bmatrix} \begin{Bmatrix} \xi_s \\ \xi_f \end{Bmatrix} = \begin{Bmatrix} \Phi_s^T F_s \\ \Phi_f^T F_f \end{Bmatrix} \quad (2-85)$$

This modal transient formation 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 [Equation \(2-77\)](#) 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 [Equation \(2-74\)](#) for structural problems. Due to the similarities of their equations, the solution process is again identical.

Adjoint Sensitivity Analysis for Grid Responses

As detailed above, the criterion for invoking the adjoint sensitivity method is

$$NRESP < NDV \cdot NLC \quad (2-86)$$

The actual selection of the adjoint sensitivity method involves further factors/restrictions:

- The adjoint method is only available for statics (not including thermal loads or static aeroelasticity) and frequency response analyses (both direct and modal).
- Only grid responses are supported. SPC Forces are not supported.
- The method is not available with p-element models

The limitation to grid responses is motivated by two factors: 1) The development of the $\partial f / \partial u$ vector required in [Equation \(2-33\)](#) is not trivial for certain elements and 2) it is unlikely that a design task that included element responses would pass the criterion of [Equation \(2-86\)](#).

The adjoint sensitivity analysis shares the property or shape basic vector perturbation step detailed by [Equation \(2-40\)](#) thru [Equation \(2-45\)](#) with the direct method, but the two methods diverge at that point. The adjoint method requires solution of the adjoint equation

$$[S]^T \{\lambda\} = \left\{ \frac{\partial f}{\partial u} \right\} \quad (2-87)$$

where the $\{\partial f / \partial u\}$ vectors are established based on the retained grid response(s) and the rows of these vectors are all zeroes except for a 1.0 in the location corresponding to the degree-of-freedom of the active response. The sensitivity can then be developed directly using

$$\left\{ \frac{\partial r}{\partial x} \right\} = -\{\lambda\}^T [-\omega^2 [\Delta M] + i\omega [\Delta B] + [\Delta K]] \{u\} \quad (2-88)$$

where the terms on the right-hand side are developed using the semi-analytic methods similar to those given in [Equation \(2-47\)](#) and [Equation \(2-48\)](#).



Because many of the steps required in the direct sensitivity analysis are not needed in the adjoint method, this permits additional savings in terms of CPU time and required disk space over and above the savings produced from the fewer number of adjoint loads relative to the number of pseudo-loads.

[Equation \(2-87\)](#) is similar to the equation solved in a frequency response analysis. This similarity can be exploited by solving for the adjoint solution vectors of $\{\lambda\}$ as part of the analysis. This is particularly useful in a direct frequency response analysis where a considerable portion of the time can be expended in the decomposition of the $[S]$ matrix at each frequency. See the description of PARAM SOLADJC in [SOLADJC](#) for details as to when and how the adjoint solution vectors are computed during the frequency response analysis.

This adjoint discussion has focused on frequency response analysis, but adjoint methods are also available for static and static aeroelastic analysis. In most cases, the restriction on element responses as outlined above precludes the use of adjoint methods in a statics analysis, but there is an important application that relies on the adjoint method: compliance responses. Compliance is defined as the dot product of the load times the displacement.

This response type is central in topology optimization where the number of design variables can be in the thousands. The adjoint solution vector is equal to structural solution vector so there is no need to recalculate this and the sensitivity becomes:

$$\left\{ \frac{\partial r}{\partial x} \right\} = -\{u\}^T [\Delta K] \{u\} \quad (2-89)$$

If the load is a function of the design variable, as is the case with the design of loads in [Table 2-4](#), an extra term is provided to give:

$$\left\{ \frac{\partial r}{\partial x} \right\} = 2\{u\}^T \{[[\Delta P] - [\Delta K]]\{u\}\} \quad (2-90)$$

Weight Sensitivities

Weight sensitivities are calculated using matrix $[\Delta M]$ and matrix $[DR_g]$. The latter matrix provides the displacement at all the degrees-of-freedom in the model based on a unit displacement at the six degrees-of-freedom at the grid identified by PARAM GRDPNT.

$$[\Delta W] = [DR_g]^T [\Delta M] [DR_g] \quad (2-91)$$

This is a six by six matrix for each design variable and the appropriate term in the matrix is selected based on the user's specification of the weight response.

Volume Sensitivities

Volume sensitivities are computed by computing the volume of all the finite elements for a perturbation in each of the design variables, subtracting out the baseline volume and dividing by the perturbation.

$$\frac{\Delta V}{\Delta x_i} = \frac{V(\mathbf{x}^0 + \Delta x_i) - V(\mathbf{x}^0)}{\Delta x_i} \quad (2-92)$$



Weight as a function of Material and Property ID Sensitivities

The sensitivity for the weight as a function of material response is similar to the volume response given above except that:

1. The elements need to be separated by material ID.
2. The volumes need to be multiplied by the density associated with the material.

For the weight of the property ID, MSC Nastran already has the ability to provide that weight for each property ID so again, a finite difference calculation is used to compute the sensitivity.

For composites the calculations are more involved since the layered composites have been converted to anisotropic materials with spawned materials and thicknesses. The original PCOMP/G information has been retained so the total weight of each element needs to be broken down to give the weight of the material/property id of interest.

Real Eigenvalue Sensitivities

The eigenvalue equation is

$$([K] - \lambda_n[M])\{\phi_n\} = 0 \quad (2-93)$$

where λ_n and $\{\phi_n\}$ are the n -th eigenvalue and eigenvector, respectively. $[K]$ is the structural stiffness, and $[M]$ is the structural mass.

The governing [Equation \(2-93\)](#) can be differentiated with respect to the i -th design variable x_i to yield,

$$([K] - \lambda_n[M]) \frac{\partial \{\phi_n\}}{\partial x_i} + \left(\frac{\partial [K]}{\partial x_i} - \lambda_n \frac{\partial [M]}{\partial x_i} \right) \{\phi_n\} = \frac{\partial \lambda_n}{\partial x_i} [M] \{\phi_n\} \quad (2-94)$$

When [Equation \(2-94\)](#) is pre-multiplied $\{\phi_n\}^T$, the $\{\phi_n\}^T ([K] - \lambda_n[M])$ term that then appears at the front of [Equation \(2-94\)](#) is the transpose of [Equation \(2-93\)](#) and therefore zero. [Equation \(2-94\)](#) can then be solved for the eigenvalue derivatives

$$\frac{\partial \lambda_n}{\partial x_i} = \frac{\{\phi_n\}^T \left(\frac{\partial [K]}{\partial x_i} - \lambda_n \frac{\partial [M]}{\partial x_i} \right) \{\phi_n\}}{\{\phi_n\}^T [M] \{\phi_n\}} \quad (2-95)$$

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 [Equation \(2-47\)](#) and [Equation \(2-48\)](#). [Equation \(2-95\)](#) is solved for each retained eigenvalue referenced in the design model and for each design variable. It should be mentioned that this equation is valid only for the case of distinct eigenvalues.

Real Eigenvector Sensitivities

The eigenvector sensitivity method available in MSC Nastran is based on Nelson's Method (see Nelson, R. B., in Reference 4.. For the method, [Equation \(2-94\)](#) can be rewritten as



$$([K] - \lambda_n[M]) \frac{\partial \{\phi_n\}}{\partial x_i} = -\left(\frac{\partial [K]}{\partial x_i} - \lambda_n \frac{\partial [M]}{\partial x_i} + \frac{\partial \lambda_n}{\partial x_i} [M] \right) \{\phi_n\} \quad (2-96)$$

Equation (2-96) cannot be solved directly for the eigenvector sensitivity because the $([K] - \lambda_n[M])$ term is singular. This requires that the matrix size be reduced by one and the reduction technique differs depending on what method the user has specified for the normalization of the eigenvector. If the NORM=MAX option has been selected, then the maximum term is assumed invariant so that this term can be removed from the system, the reduced system solved, and a zero placed in the location of the maximum term.

For NORM=MASS, a substitution is made

$$\left\{ \frac{\partial \phi_n}{\partial x_i} \right\} = \{V_i\} + C_{in}\{\phi_n\} \quad (2-97)$$

Without loss of generality, one of the terms in V_i can be set to 0.0 and this row and column can be partitioned out of the system to produce a reduced system than can be designated with a superscript bar ($\bar{\cdot}$)

$$[\bar{k}] - \lambda_i[\bar{M}]\{\bar{V}_{\bar{n}}\} = -\left(\frac{\partial [\bar{K}]}{\partial x_i} - \lambda_n \frac{\partial [\bar{M}]}{\partial x_i} + \frac{\partial \lambda_n}{\partial x_i} [\bar{M}] \right) \{\bar{\phi}_n\} \quad (2-98)$$

Equation (2-97) contains a C_{in} term that can be derived from the requirement that the generalized mass is invariant with respect to the design change

$$2\{\phi_n\}^T [M] \frac{\partial \phi_n}{\partial x_i} + \phi_n^T \left[\frac{\partial M}{\partial x_i} \right] \phi_n = 0 \quad (2-99)$$

Substituting **Equation (2-97)** into **Equation (2-99)** yields

$$C_{in} = -\frac{1}{2} \left(2\{\phi_n\}^T [M] \{V_n\} + \{\phi_n\}^T \left[\frac{\partial M}{\partial x_i} \right] \{\phi_n\} \right) \quad (2-100)$$

The total sensitivity for the NORM=MASS method can now be determined by substituting results from **Equation (2-98)** and **Equation (2-100)** into **Equation (2-97)**.

It should be noted that NORM=POINT is not supported for eigenvector sensitivity analysis.

Buckling Load Factor Sensitivities

The derivation of buckling sensitivity is similar to that of eigenvalue sensitivities. The equation for determining of the buckling load factor λ is

$$[K]\{\phi_n\} + \lambda_n[K_d]\{\phi_n\} = 0 \quad (2-101)$$

Differentiating this expression and solving for the buckling load factor sensitivities yields



$$\frac{\partial \lambda_n}{\partial x_i} = \frac{\{\phi_n\}^T \left(\frac{\partial [K]}{\partial x_i} + \lambda_n \frac{\partial [K_d]}{\partial x_i} \right) \{\phi_n\}}{\{\phi_n\}^T [K_d] \{\phi_n\}} \quad (2-102)$$

As before, this equation is solved semi analytically by approximating the structural property matrix derivatives using finite differences.

The approximation of the differential stiffness is worth discussing. This stiffness is a function not only of geometry but of displacement as well

$$[K_d] = [K_d(\mathbf{X}, \mathbf{U})] \quad (2-103)$$

where the displacement is of a particular static analysis that is used in the buckling analysis.

In the finite difference approximation, $[K_d]$ can be approximated for the perturbed configuration as

$$\frac{\partial [K_d]}{\partial x_i} \cong \frac{[K_d(\mathbf{X}^0 + \Delta x_i, \mathbf{U} + \Delta \mathbf{U})] - [K_d(\mathbf{X}^0, \mathbf{U})]}{\Delta x_i} \quad (2-104)$$

MSC Nastran supports two variations for the sensitivity of the differential stiffness, neither of which is exactly of the form given in (2-104). In the first, the differential stiffness matrix is ignored in its entirety. In some situations, this provides an adequate approximation to the buckling sensitivity and certainly requires less computational resources. The second alternative is to include the sensitivity of the differential stiffness using

$$\frac{\partial [K_d]}{\partial x_i} = [K_d(\mathbf{X}^0 + \Delta x_i, \mathbf{U})] - [K_d(\mathbf{X}^0, \mathbf{U})] + \left[\frac{[K_d(\mathbf{X}^0, \Delta \mathbf{U})]}{\Delta x_i} \right] \quad (2-105)$$

This form relies on the assumption that the differential stiffness matrix is a linear function in the displacements so that

$$[K_d(\mathbf{x}^0, \mathbf{U} + \Delta \mathbf{U})] - [K_d(\mathbf{x}^0, \mathbf{U})] \cong [K_d(\mathbf{x}^0, \Delta \mathbf{U})] \quad (2-106)$$

Experience has shown that this is a more accurate formulation than using differencing techniques to obtain the effects of the perturbed displacements. The selection of which method is applied in a buckling sensitivity calculation is based on PARAM DSNOKD. The default value for this parameter is 0.0, which neglects the differential stiffness sensitivity. In the case where the analysis is being carried out on a determinant structure, the differential stiffness matrix is invariant with respect to property changes so neglecting this term is appropriate. DSNOKD=1.0 uses the formulation of Equation (2-105) and is recommended when accuracy is important.

Complex Eigenvalue Sensitivity Analysis

The basic equation for complex eigenvalue analysis is

$$([M]\mathbf{p}_n^2 + [\mathbf{B}]\mathbf{p}_n + [\mathbf{K}])\{\phi_n\} = 0 \quad (2-107)$$

and there is an associated left-handed eigenvalue analysis



$$([M]^T p_n^2 + [B]^T p_n + [K]^T) \{ \psi_n \} = 0 \quad (2-108)$$

where p_n , ϕ_n and ψ_n denote the complex eigenvalue and right and left complex eigenvectors, respectively, of the n th mode. Note that the left and right eigenvectors are distinct and share a common eigenvalue, p_n which can be defined as

$$p_n = \alpha + i\omega \quad (2-109)$$

where α and ω are the real and imaginary parts of the complex eigenvalue, p_n . The structural matrices $[K]$, $[B]$, and $[M]$ include contributions both from structural elements as well as direct input via K2GG, B2GG, M2GG, K2PP, B2PP, and M2PP.

Relations used in computing the design sensitivities can be derived by differentiating [Equation \(2-107\)](#) and [Equation \(2-108\)](#) with respect to a design variable. Using Δ to refer to a perturbation, we obtain, from [Equation \(2-107\)](#),

$$([M + \Delta M](p + \Delta p)_n^2 + [B + \Delta B](p + \Delta p)_n + [K + \Delta K]) \{ (\phi + \Delta \phi)_n \} = 0 \quad (2-110)$$

Expanding, ignoring higher order terms, and utilizing [Equation \(2-108\)](#) yields the following approximation for the variation in the n -th complex eigenvalue

$$\Delta p_n \cong \frac{-\{\psi_n\}^T ([\Delta M]p_n^2 + [\Delta B]p_n + [\Delta K]) \{\phi_n\}}{\{\psi_n\}^T (2p_n[M] + [B]) \{\phi_n\}} \quad (2-111)$$

where

$$\Delta p_n = \Delta \alpha + i \Delta \omega \quad (2-112)$$

and $\Delta \alpha$ and $\Delta \omega$ are the real and imaginary variations, respectively, of the complex eigenvalue. Note that the approximation in [Equation \(2-111\)](#) does not require complex eigenvector sensitivities. As in the case of modal dynamic analysis of [Equation \(2-73\)](#) and [Equation \(2-46\)](#), the assumption is that enough modes have been retained such that the perturbed subspace can be adequately represented by the original mode set. For this reason, a larger mode set than would normally be required for analysis purposes is recommended for modal complex sensitivity analysis. Because this number is highly problem-dependent, a logical approach would be to include enough modes until acceptable accuracy convergence is achieved.

Flutter Sensitivity Analysis

The eigenvalue problem for flutter adds aerodynamic terms to the derivation given for complex eigenanalysis in the previous section. The flutter equation is given by

$$\left[M_{hh} p^2 + \left(B_{hh} - \frac{1}{4} \rho \bar{c} V \frac{Q_{hh}^l}{k} \right) p + \left(K_{hh} - \frac{1}{2} \rho V^2 Q_{hh}^R \right) \right] \{ u_h \} = 0 \quad (2-113)$$

where the terms in the above equation are defined in the [Flutter Solution Techniques](#) in Chapter 2 of the *MSC 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

$$p = \omega(\gamma + i) = p_R + p_I \quad (2-114)$$

The sensitivity calculation for $\Delta\gamma$ is expressed in terms of the sensitivities of the eigenvalue as

$$\frac{\partial\gamma}{\partial x} = \frac{1}{p_I} \left(\frac{\partial p_R}{\partial x} - \frac{\gamma \partial p_I}{\partial x} \right) \quad (2-115)$$

The eigenvalue sensitivity computation is conceptually straightforward and somewhat along the lines of the complex eigenvalue sensitivity given above. The complete derivation can be found in [Aeroelastic Design Sensitivities and Optimization](#) in the *MSC Nastran Aeroelastic Analysis User's Guide*.

PSD Response Sensitivities

The calculation of the sensitivities of the PSD responses of [Equation \(2-18\)](#) is based on sensitivities calculated from a dynamic sensitivity analysis

$$\Delta S_{jf}^x = \sum_a \sum_b S_{ab} (H_{ja} \Delta H_{jb}^* + \Delta H_{ja} H_{jb}^*) \quad (2-116)$$

where the terms ΔH are sensitivities at a particular degree-of-freedom and frequency.

RMS Response Sensitivities

The sensitivity of the RMS quantity \bar{u}_j versus each design variable x is calculated from a numerical integration of the PSD sensitivities of [Equation \(2-116\)](#) and the definition of RMS response given by [Equation \(2-20\)](#)

$$\Delta \bar{u}_j = \frac{1}{2} \left(\frac{1}{\bar{u}_j} \right) \left[\sum_f C_f (\Delta S_{jf}) \right] \quad (2-117)$$

Sparse Data Recovery in Sensitivity Analysis

A final aspect of sensitivity analysis bears mentioning simply because of the dramatic effect it can have on performance and data storage requirements. As finite element model sizes become even larger, it has become apparent that there is a benefit to be obtained in only computing minimum required values of physical displacements. Data recovery refers to the steps that occur following the solution of equations like [Equation \(2-34\)](#) in physical coordinates or [Equation \(2-71\)](#) in model coordinates. The concept of sparse data recovery will be explained here in the modal context, since it is here that the most dramatic gain can be achieved. The recovery from modal to physical coordinates occurs with the matrix multiply given in [Equation \(2-70\)](#). Sparse data recovery entails limiting the terms in the $[\Phi]$ matrix to those that are to be used subsequently. In the context of design sensitivity, the total degrees-of-freedom required in a sensitivity analysis are a union of sets defined by

- The user output requests.
- Those required to evaluate the design responses.



- Those touched by the design model and needed either to compute the pseudo-loads or in the sensitivity calculations of [Equation \(2-88\)](#).

In extreme cases, a finite element model may have millions of degrees-of-freedom while only a few hundred need to be recovered.

Sparse data recovery is currently employed in three analysis disciplines in SOL 200:

1. Normal Modes
2. Direct Frequency Analysis
3. Modal Frequency Analysis

While other disciplines could be added to this list, the three listed above are considered the most important.



Optimization

A variety of optimization algorithms are available for use in SOL 200. This subchapter is limited to a discussion of the interaction between the optimizer and the approximate model and is applicable regardless of the selected optimization algorithm. An overview of the available optimization algorithms is given in [Optimizer, 30](#) while the licensing considerations are discussed in [Optimizers \(Licensing\), 133](#).

Connection Between the Optimizer and the Approximate Model

[Figure 2-7](#) is a flow chart of the approximate optimum design task using the specified optimization algorithm as it is embedded within MSC Nastran. Note that the optimizer is called within a ‘do-while’ loop and that, based on a the INFO parameter, the approximate model discussed in [The Approximate Model](#) is used to provide function values or gradient values. The figure shows the possibility of an error being encountered by the optimizer that makes further progress impossible. This is, hopefully, a rare occurrence that is triggered by, for example, attempting an optimization task that is too big or asking the optimizer to overcome a violated constraint for which the gradient is zero. In this latter case, there is no possibility of achieving a feasible design and this usually indicates a user error in specifying the design task.

As seen in [Figure 2-7](#), if INFO=2, a gradient evaluation is required. This is a request for sensitivities of the objectives and all the constraints the optimizer considers active. If INFO is 0 or 1, a function evaluation is required. This is a request for the value of the objective and all of the constraints for the current values of the design variables. INFO=0 indicates that the design task is complete, the ‘do-while’ loop is exited and control is returned to the MSC Nastran system.



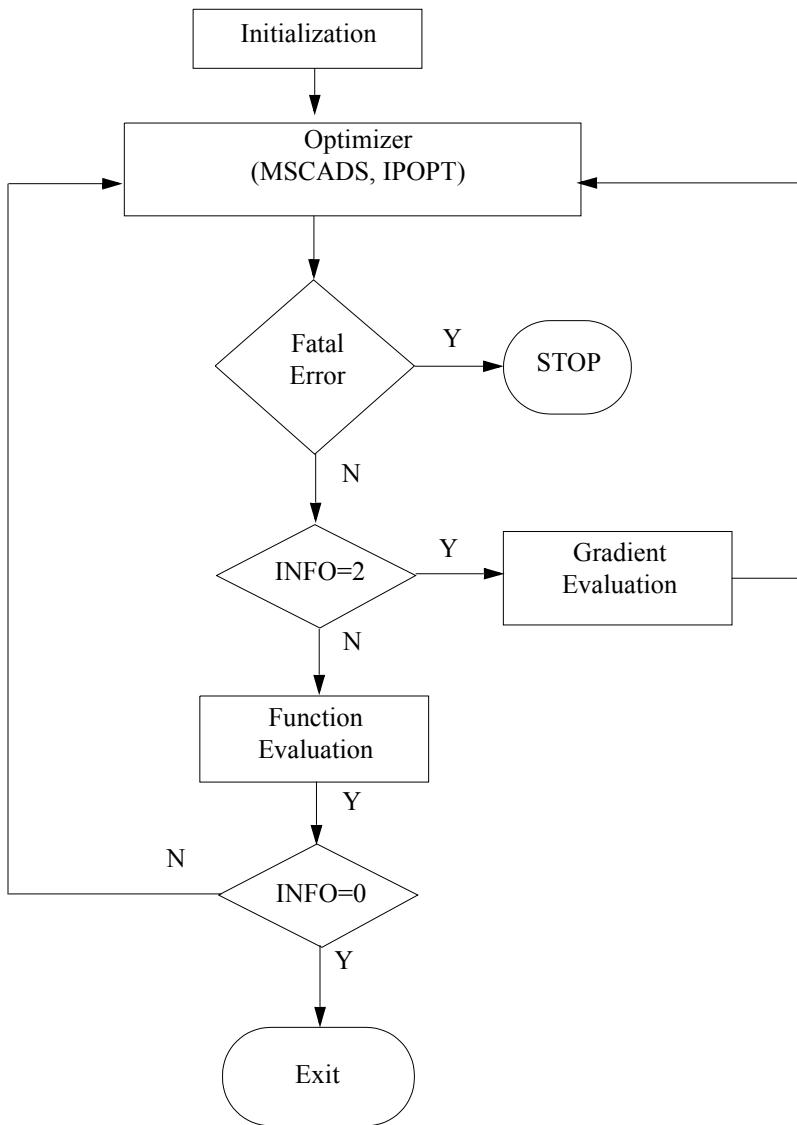


Figure 2-7 Interaction Between the Optimizer and the Approximate Function and Gradient Evaluations

Numerically Identifying the Active and Violated Constraints

The optimization algorithm makes a determination as to which of the retained constraints are violated and which are active. The constraints which are neither active nor violated can be ignored in the gradient



evaluation of [Figure 2-7](#), thereby reducing the amount of computations required, the amount of computer memory required and the size of the mathematical programming task. [Figure 2-8](#) illustrates the concept of active and violated constraints for a single inequality constraint in a simple two-design variable space and [Figure 2-9](#) presents the same information in a plot of a constraint value as a function of a single design variable or search direction. A constraint is considered active if its numerical value exceeds CT. The default value for CT is -0.03, but can be changed by the user. Once a constraint is active, its gradient is included in the search direction computation. An active constraint may subsequently become inactive if its value falls below CT.

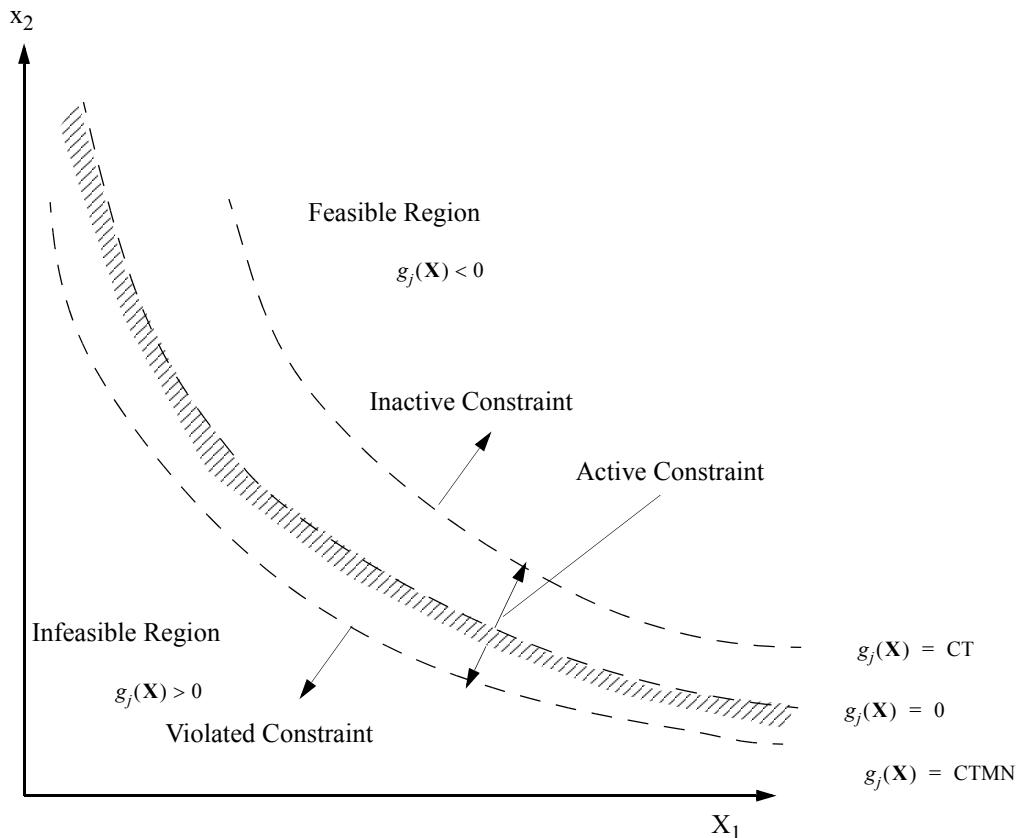


Figure 2-8 Active and Violated Constraints



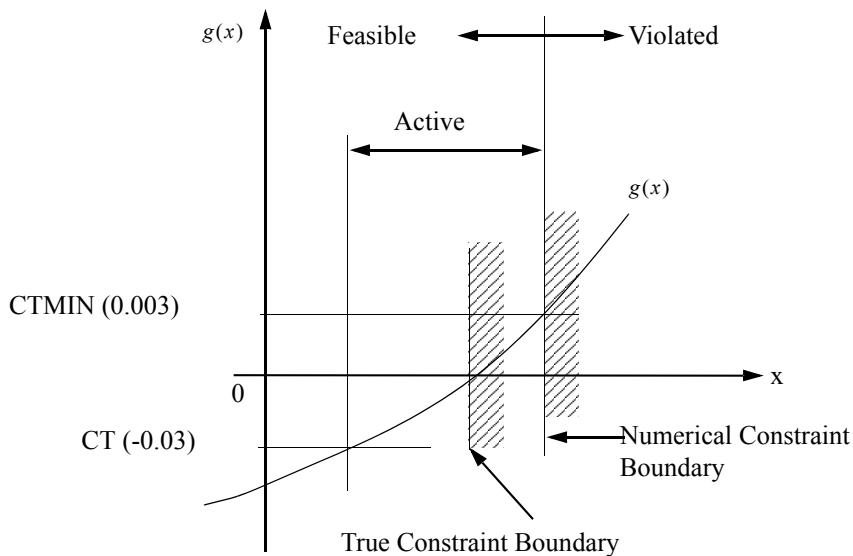


Figure 2-9 CT and CTMIN

The optimizer designates a constraint as violated if its value is greater than CTMIN. The CTMIN default is 0.003 as is seen in [Figure 2-9](#). Thus, some small constraint violation (three-tenths of one percent, by default) is tolerated.



The Approximate Model

MSC Nastran employs a number of techniques, collectively referred to as *Approximation Concepts*, to make design optimization possible for large finite element models. Design variable linking, reduced basis vectors, constraint screening and load case deletion have already been described and these approximation concepts all play a role in reducing the cost of the design task. This subsection deals with the concept of the Approximate Model that can be thought of as enabling design optimization as it is performed in MSC Nastran. MSC Nastran uses formal approximations to the finite element analysis and the sensitivity analysis to avoid the high cost associated with repeated finite element analyses during design optimization. As shown in [Figure 2-7](#), the optimizer interacts with the approximate model when it requires information. Given a set of design variables, the optimizer needs information on the function values (the value of the objective and the values of the retained constraints) and the gradient values (gradient of the objective with respect to the design variables and the gradient of the active constraints with respect to the design variables). This section first introduces the concept of using Taylor Series expansion to approximate the finite element analysis and follows this with a discussion of how move limits are applied to try to restrict the changes in the design model to a region where the approximations are valid. With this background, it is possible to describe the Function Evaluations and Gradient Evaluations of [Figure 2-7](#).

Formal Approximations

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

$$f(\mathbf{X}^0 + \Delta x) = f(\mathbf{X}^0) + \frac{df}{dx}\Bigg|_{\mathbf{X}^0} \cdot \Delta x + \frac{d^2f}{dx^2}\Bigg|_{\mathbf{X}^0} \cdot \frac{\Delta x^2}{2!} + \frac{d^3f}{dx^3}\Bigg|_{\mathbf{X}^0} \cdot \frac{\Delta x^3}{3!} + \dots \quad (2-118)$$

In addition to the function value $f(\mathbf{X}^0)$, this series requires that all derivatives at \mathbf{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.

In MSC Nastran, we only include through the first derivative term in the series to obtain a linear approximation

$$\tilde{f}(\mathbf{X}^0 + \Delta x) = f(\mathbf{X}^0) + \frac{df}{dx}\Bigg|_{\mathbf{X}^0} \cdot \Delta x \quad (2-119)$$

Since all terms of power and higher have been omitted, the error in the approximation is on the order of Δx^2 . This situation is shown in [Figure 2-10](#) where the error increases with increasing values of $|\Delta x|$. Note that the approximation would be exact if the original function were linear.



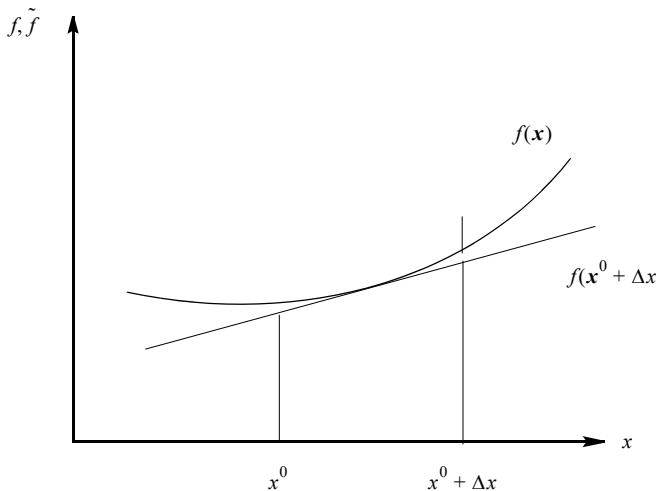


Figure 2-10 Errors in Approximating Functions

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

$$\begin{aligned}\tilde{F}(\mathbf{X}^0 + \Delta\mathbf{x}) &= F(\mathbf{X}^0) + (\nabla F)_{\mathbf{X}^0} \cdot \Delta\mathbf{x} \\ g_j(\mathbf{X}^0 + \Delta\mathbf{x}) &= g_j(\mathbf{X}^0) + (\nabla g_j)_{\mathbf{X}^0} \cdot \Delta\mathbf{x}\end{aligned}\quad (2-120)$$

where a gradient term replaces the first derivative term of [Equation \(2-119\)](#).

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 refer to the [A Simple Structural Example](#).

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 [Equation \(2-120\)](#), we have

$$\begin{aligned}\tilde{V}(B^0 + \Delta B, H^0 + \Delta H, L) &= V(B^0, H^0, L) + \frac{\partial V}{\partial B} \Big|_{B^0, H^0} \cdot \Delta B + \frac{\partial V}{\partial H} \Big|_{B^0, H^0} \cdot \Delta H \\ \tilde{\sigma}(B^0 + \Delta B, H^0 + \Delta H, L) &= \sigma(B^0, H^0, L) + \frac{\partial \sigma}{\partial B} \Big|_{B^0, H^0} \cdot \Delta B + \frac{\partial \sigma}{\partial H} \Big|_{B^0, H^0} \cdot \Delta H \\ \tilde{\delta}(B^0 + \Delta B, H^0 + \Delta H, L) &= \delta(B^0, H^0, L) + \frac{\partial \delta}{\partial B} \Big|_{B^0, H^0} \cdot \Delta B + \frac{\partial \delta}{\partial H} \Big|_{B^0, H^0} \cdot \Delta H\end{aligned}\quad (2-121)$$



For an initial design at ($B = 6$, $H = 45$), the derivatives in the above equation can be evaluated based on [Equation \(1-15\)](#) through [Equation \(1-17\)](#) to yield

$$\begin{aligned}\tilde{V}(B^0 + \Delta B, H^0 + \Delta H, L) &= 1.35 \times 10^5 + 2.25 \times 10^4 \Delta B + 3.0 \times 10^3 \Delta H \\ \tilde{\sigma}(B^0 + \Delta B, H^0 + \Delta H, L) &= 555.56 - 92.593 \Delta B - 24.691 \Delta H \\ \tilde{\delta}(B^0 + \Delta B, H^0 + \Delta H, L) &= 2.0576 - 0.34294 \Delta B - 0.13717 \Delta H\end{aligned}\quad (2-122)$$

The design space resulting from [Equation \(2-122\)](#) is shown in [Figure 2-11](#), 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.

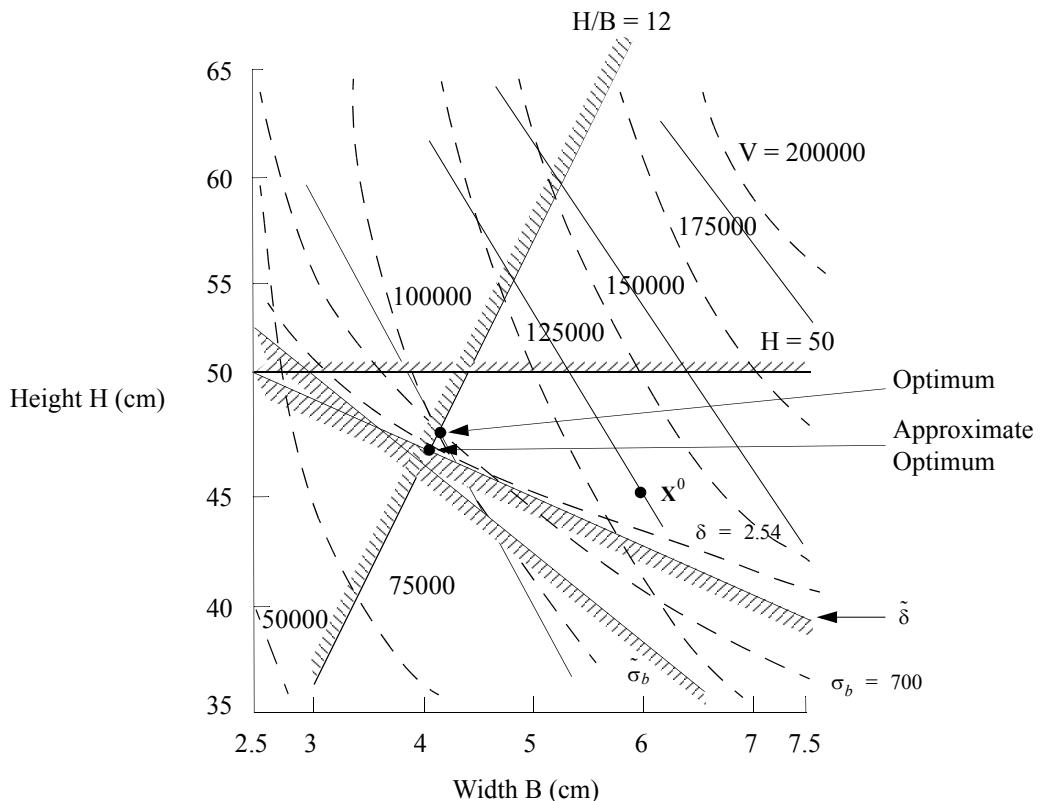


Figure 2-11 Linearly Approximated Cantilever Beam Design Space

Though not yet an optimal design, the approximate optimum is a useful starting point for the next approximate optimization cycle.

Move Limits

Since the approximations just discussed are only locally valid, limits are placed on the amount by which the design is allowed to vary during the approximate optimization. In MSC Nastran, move limits are imposed in terms of both allowable changes in design variables and allowable changes in properties.

Design Variable Limits

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 (Equation (2-5)). 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 can also be included.

During approximate optimization, each design variable is bounded from above and below by

$$x_i^L \leq x_i \leq x_i^U \quad (2-123)$$

x_i^L = the lower bound on the i-th design variable

x_i^U = the upper bound

These bounds are determined at the beginning of each design cycle according to

$$\begin{aligned} x_i^L &= \text{MAX}(XLB, x_i^0 - \text{MOVE}) \\ x_i^U &= \text{MIN}(XUB, x_i^0 + \text{MOVE}) \end{aligned} \quad (2-124)$$

MOVE = MAX(DXMIN, ($|x_i| \cdot \text{DELXV}$))

XLB = the lower bound on the i-th design variable

XUB = the upper bound

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 0.5 or 50% (0.2 or 20% for topology optimization) change in all design variables. The DXMIN value ensures that the move limits don't become too restrictive when the design variable value is near zero. The default value for DXMIN is 0.05 (1.0E-5 for topology optimization), but it, too, can be modified using the DOPTPRM entry.



Property Limits

Move limits can be imposed on designed properties as well, but not all properties. [Designed Properties](#) has discussed the three types of properties: type-1, type-2, and type-3. As shown there, type-1 properties are linear in terms of the design variables (see [Equation \(2-2\)](#))

$$p_j = c_o + \sum_i c_i x_i$$

while type-2 enable a nonlinear relationship between design variables and properties (see [Equation \(2-5\)](#))

$$p_j = f(\mathbf{X}, \mathbf{C})$$

and type-3 refers to beam properties that are spawned when using the beam library. Move limits are always imposed on type-2 and type-3 properties¹ but they are not imposed on type-1 properties when the property is controlled by a single design variable

$$p_j = C_o + c_i x_i \quad (2-125)$$

and the user has not imposed bounds on the value of the property.

Move limits are not imposed in this case since the move limit on the design variable already restricts the change in the property value, i.e., if the DELXV value of [\(2-124\)](#) imposes a 50% change on the i-th design variable of, the p_j value can never exceed 50% on a given design cycle. Not imposing limits on these properties simplifies the design optimization task by reducing the number of constraints that have to be considered. You should not impose bounds on properties that satisfy equation 2-125 unless there is a clear reason to do so.

For the type-1 properties that are a function of more than one design variable and/or that have user imposed limits and for type-2 or type-3 properties, bounds are imposed from below and above the designed properties by

$$p_j^L \leq p_j \leq p_j^U \quad (2-126)$$

p_j^L = the lower bound on the j-th property

p_j^U = 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

$$\begin{aligned} p_j^L &= \text{MAX(PMIN, } p_j^0 - \text{PMOVE)} \\ p_j^U &= \text{MIN(PMAX, } p_j^0 + \text{PMOVE)} \end{aligned} \quad (2-127)$$

¹Move limits are not placed on type-2 and type-3 properties that can change sign. Placing move limits on properties that can be zero has been shown to create problems



PMOVE = MAX(DPMIN, $p_j^0 \cdot \text{DELP}$)

PMIN = the lower bound on the property

PMAX = the upper bound

DELP = the maximum allowable percentage change in the property

DELP can be defined on the DOPTPRM entry and has a default of 0.2 for a 20% maximum property variation. DPMIN can also be defined on the DOPTPRM entry and has a default value of 0.01.

Figure 2-12 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 Equation (2-127). 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. Similarly, PMAX will become the upper bound if it is less than p_j^U .

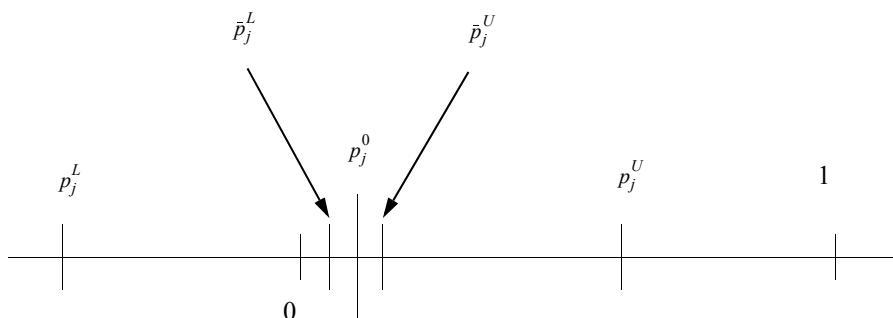


Figure 2-12 Move Limits on Properties

Equation (2-124) and Equation (2-127) effectively form a “box” around the current design. This effect is shown in Figure 2-13 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 2-13 is merely to suggest some general features of the overall process.



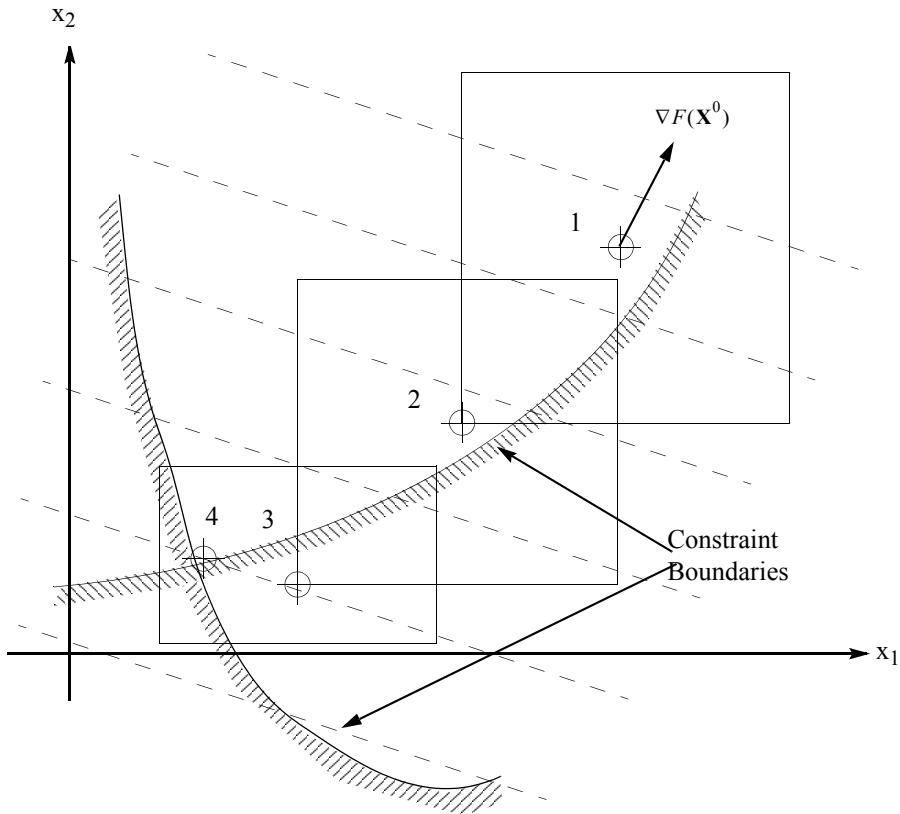


Figure 2-13 Sequence of Approximations

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, the code may automatically adjust these move limits if the design task is performing poorly. 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. Discrepancies have thus been detected between the approximate and the true responses.

If these discrepancies continue 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 MSC Nastran if the following criteria are satisfied:

1. The current design cycle number is greater than or equal to three.
2. 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 [Modification of Move Limit Parameters](#)). If the job is to be restarted, it is recommended that an updated DOPTPRM entry with the new move limits be included in the restart file.

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 [\(2-124\)](#) are simply input directly to the optimizer.

Move limits that are imposed on properties are implemented as equivalent constraints. For the j -th property in the design model, the equivalent lower and upper bound constraints are given by

$$g_L(\mathbf{X}) = \frac{p_j^L - p_j(\mathbf{X})}{|p_j^L|} \leq 0 \quad (2-128)$$

$$g_U(\mathbf{X}) = \frac{p_j(\mathbf{X}) - p_j^U}{|p_j^U|} \leq 0$$

where the lower and upper bounds p_j^L and p_j^U are given by [Equation \(2-127\)](#).

Transformation of the Approximate Optimization Task to a Feasible Design

This section addresses an optional feature of the approximate optimization task. A general statement can be made that optimization algorithms perform best when the design task does not include violated constraints. In fact, some algorithms will fail if a feasible design does not exist (the algorithms used in MSC Nastran search for the best compromise infeasible design so that the transformation to a feasible design is not a strict requirement). In order to facilitate the use of a general optimization procedure, a simple transformation technique has been developed to ensure that the optimization task always works in the feasible domain.



The standard optimization task minimizes an objective subject to satisfying a set of constraints. The transformed problem, designated the β Method, involves creating a β design variable that modifies the constraints so that:

$$g_j' = g_j - \beta \quad (j = 1, 2, \dots, ncon)$$

and transforms the objective function so that:

$$\Phi' = \Phi + \beta * \text{PENAL} * \Phi_0$$

where PENAL is the user defined penalty parameter, Φ_0 is the initial objective function and β is initialized to the maximum initial constraint value. (If this maximum value is less than CTRMIN, the transformation is not applied.) In this way, the maximum constraint value is never positive while the penalty on the objective acts to force the β value to zero. If there is a feasible design, this technique should lead to it. If there is not, this will result in a compromise design where the maximum constraint violation is minimized.

Function Evaluation

The function evaluation proceeds in five steps:

1. The properties are evaluated exactly.
2. Approximate type-1 responses are evaluated.
3. Approximate type-2 responses are evaluated.
4. Approximate type-3 responses are evaluated.
5. The the objective and the constraints are evaluated .

Not all these steps need to be present in a particular design task. Each of these steps is now described.

Property Evaluation

Given the design variable values, the properties can be computed directly and exactly. For type-1 properties that are a linear function of the design variables, the properties are evaluated using a multiply and add operation (see [Equation \(2-2\)](#)). For type-2 properties, equations that contain constants plus design variable values need to be evaluated on an individual basis. If beam library dimensions are being designed, it is necessary to compute the beam properties. Some of these properties are linear in the design variables and can be computed in a fashion similar to the type-1 properties. Others are nonlinear, and it is necessary to call the beam library routines to calculate these properties.

Type-1 Response Evaluation

Three basic types of approximation are available in the computation of type-1 responses:

- Direct variable approximations.
- Reciprocal variable approximations
- Convex linearization



The method used is based on the user specified parameter APRCOD that is discussed later with the [DOPTPRM](#) entry.

The formal approximations used in MSC Nastran are based on simple first-order Taylor series expansions. The general form of this expansion for a response is

$$r_j(\mathbf{X}^0 + \Delta\mathbf{x}) = r_j(\mathbf{X}^0) + \sum_i \frac{\partial r_j}{\partial x_i} \Big|_{\mathbf{X}^0} \cdot \Delta x_i \quad (2-129)$$

The required derivative information is available from the design sensitivity analysis as outlined in [Design Sensitivity Analysis](#). Note that the approximate response, $r_j(\mathbf{x}^0 + \Delta\mathbf{x})$, in [Equation \(2-129\)](#) is linear in the design variable change $\Delta\mathbf{x}$. In some instances, the response actually is a linear function of the design variable and [Equation \(2-129\)](#) provides an exact response value. In general, however, we expect some error when we try to approximate responses that are actually nonlinear in $\Delta\mathbf{x}$.

Direct Approximations

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

$$r_j^D(\mathbf{X}^0 + \Delta\mathbf{x}) = r_j(\mathbf{X}^0) + \sum_i \frac{\partial r_j}{\partial x_i} \Big|_{\mathbf{X}^0} \cdot \Delta x_i \quad (2-130)$$

where the superscript D indicates a direct approximation.

The quantity $\Delta\mathbf{x}$ represents a total vector move in the design space from the initial design \mathbf{X}^0 . The partial derivatives $\partial r_j / \partial x_i$ are available from the design sensitivity analysis.

Reciprocal Approximations

The first-order Taylor series expansions of [Equation \(2-129\)](#) 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 2-14](#) where the cross-sectional area A is taken as the design variable.

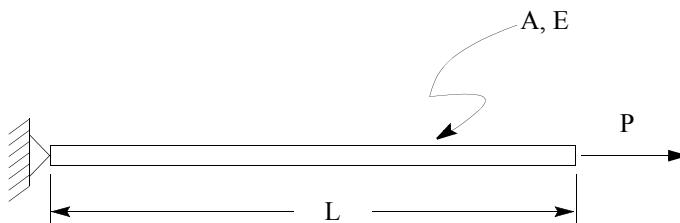


Figure 2-14 Axially Loaded Bar Element

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 I/A or I/E will produce approximations which are exact for both the stress and the displacement. (The stress is not a function of E so that while the approximation of stress as a function of E is exact, it is also trivial.) 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 (2-130), we get

$$r_j(\mathbf{Y}^0 + \Delta\mathbf{y}) = r_j(\mathbf{Y}^0) + \sum_i \frac{\partial r_j}{\partial y_i} \Bigg|_{\mathbf{Y}^0} \Delta y_i \quad (2-131)$$

Setting the intermediate variables to reciprocals of the design variables

$$y_i = \frac{1}{x_i} \quad (2-132)$$

yields

$$r_j^R(\mathbf{Y}^0 + \Delta\mathbf{x}) = r_j(\mathbf{X}^0) + \sum_i \frac{\partial r_j}{\partial x_i} \Bigg|_{\mathbf{X}^0} \cdot \frac{x_i^0}{x_i} (x_i - x_i^0) \quad (2-133)$$

where the superscript R indicates a reciprocal approximation.

Convex Linearization

This third approximation method is actually applied on the constraints, but is discussed here because it bears directly on the method used in the type-1 response calculation. 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.

The direct approximation for the j -th constraint is given in Equation (2-130) above while the reciprocal approximation is given in Equation (2-133).

Convex linearization is applied on an individual constraint/design variable basis. Thus, a combination of direct and reciprocal approximations may be made for a single constraint.

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

$$g_j^D(\mathbf{X}^0 + \Delta\mathbf{x}) - g_j^R(\mathbf{X}^0 + \Delta\mathbf{x}) = \frac{(x_i - x_i^0)^2}{x_i} \frac{\partial g_j}{\partial x_i} \Bigg|_{\mathbf{X}^0} \quad (2-134)$$

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



- If $\left. \frac{1}{x_i} \frac{\partial g_j}{\partial x_i} \right|_{\mathbf{x}^0} \geq 0$ use the direct approximation for g_i since it yields a larger (more conservative) estimate of constraint value.
- If $\left. \frac{1}{x_i} \frac{\partial g_j}{\partial x_i} \right|_{\mathbf{x}^0} < 0$ use the reciprocal approximation g_i since it yields a larger estimate of constraint value.

As stated, convex linearization is only applied to constraints. If the objective function is obtained from a type-1 response and the convex linearization method has been chosen, the objective function is calculated using a direct approximation. Also, if a type-1 response is used in a type-2 or type-3 response, a direct approximation is used for calculating the type-1 response even when convex linearization is used in the calculation of constraints directly based on type-1 response.

Special Handling for Eigenvalue/Frequency Response

Eigenvalue (and frequency) responses cannot be reliably approximated using either the direct or the reciprocal approximations. This is because the eigenvalue is sometimes dominated by stiffness terms, which would make it a linear function of a thickness design variable and sometimes by mass terms, which would make it a linear function of the inverse of the design variable. MSC Nastran provides an additional option in the approximation of eigenvalue responses that is given the name Rayleigh Quotient Approximation. In a normal response analysis, the eigenvalue can be expressed using Rayleigh's quotient

$$\lambda = \frac{\Phi^T K \Phi}{\Phi^T M \Phi} = \frac{U}{T} \quad (2-135)$$

where U is referred to as the modal strain energy or generalized stiffness and T is referred to as the modal kinetic energy or generalized mass.

Taylor series approximation to the modal strain and kinetic energies can be used to construct the eigenvalue approximation. It has been found that reciprocal variables provide a conservative approximation for the strain energies while direct variables are used for the kinetic energies

$$\tilde{U}_R = U_0 + \Phi^T \sum_{i=1}^n \frac{\partial K}{\partial x_i} (x_i - x_{0i}) \frac{x_{0i}}{x_i} \Phi \quad (2-136)$$

$$\tilde{T}_D = T_0 + \Phi^T \sum_{i=1}^n \frac{\partial M}{\partial x_i} (x_i - x_{0i}) \Phi \quad (2-137)$$

And, the eigenvalue approximation by the Rayleigh Quotient is

$$\tilde{\lambda}_{RQA} = \frac{\tilde{U}_R}{\tilde{T}_D} \quad (2-138)$$

This Rayleigh Quotient Approximation is the standard technique used in approximating eigenvalue and normal mode frequency responses. Approximations using direct and reciprocal approximations are also available, as explained in [DRESP1](#).



Type-2 Response Evaluation

Type-2 responses can be functions of design variables, designed properties, type-1 responses, grid locations, and other type-2 responses. The grid location may be a function of the design variables, so it is first necessary to calculate the grid locations using [Equation \(6-6\)](#). It is then a straightforward task to assemble the arguments required by a type-2 response and make the evaluation. When a type-2 response is a function of another type-2, it is necessary to perform the evaluations in a precise order so that the response values are available when needed.

An interesting special case occurs when all the design constraints and the objective are based on type-2 responses and these responses are not a function of the finite element responses. In this, perhaps academic, scenario the responses can be evaluated exactly and the optimization task is also exact. Therefore, if no move limits are placed on the design variables, the final optimal design can be obtained in a single design cycle.

Type-3 Response Evaluation

Type-3 evaluations are identical in form to type-2 evaluations with the exception that the type-2 evaluations are performed using the internal MSC Nastran equation evaluator while the type-3 evaluations are performed using the user supplied external program. type-3 responses cannot be a function of type-3 responses, so the recursive step of the type-2 evaluation is not needed.

MSC Nastran addresses the issue that type-3 response and gradient calculation can be expensive by supporting an “invariant gradient” concept. If this method is chosen, the type-3 response quantity is calculated in the same fashion as the type-1 response calculation of [Equation \(2-130\)](#).

Objective and Constraint Value Evaluation

The objective is one of the type-1, 2, or 3 responses so that it is available immediately once that responses have been evaluated. Constraint evaluations are more involved in that MSC Nastran considers four types of constraints:

- Constraints on responses
- Constraints on dependent design variables
- Constraints on properties
- Constraints on beam dimensions

The evaluation of each of these types of constraints is now considered in turn.

Constraints on Responses

Once the responses have been evaluated, it is a simple matter to apply [Equation \(2-10\)](#) to compute the constraints. An exception to this is when the convex linearization technique ([Convex Linearization](#)) is used. In this case, it is necessary to first determine which approximation concept provides the most conservative constraint value and to then use that method to evaluate the constraint.



Constraints on Dependent Design Variables

If there are dependent design variables, the lower and upper bounds on these variables are converted into constraints using

$$\begin{aligned} g_L^{DDV}(x_i) &= \frac{(XLB_i - x_i)}{|XLB_i|} \\ g_u^{DDV}(x_i) &= \frac{x_i - XUB_i}{|XUB_i|} \end{aligned} \quad (2-139)$$

where x_i is the current value of the i th dependent design variable and XLB_i and XUB_i are the bounds imposed on this variable.

These constraints differ from the side constraints of [Equation \(1-4\)](#) that are applied directly to the independent design variables by the optimizer.

Constraints on Properties

Property constraints are computed using the relationships given by [Equation \(2-128\)](#).

Constraints on Beam Dimensions

The beam library calculates beam properties based on beam dimensions. It is necessary to prevent combinations of dimensions from taking on values that are physically meaningless. For example, the inner radius of a tube cross section cannot exceed the outer radius. [Table 2-6](#) documents the constraints imposed on the MSC supplied cross-sections. Constraints are satisfied when they are ≤ 0.0 . (Refer to the *MSC Nastran Quick Reference Guide* for the meaning of the dimensions.)



Table 2-6 Constraints for Beam Library

Section Type	Constraint
TUBE	DIM2 - DIM1
I	DIM4 - DIM2
	DIM4 - DIM3
	DIM5 +DIM6 - DIM1
CHAN	2 * DIM4 - DIM2
	DIM3 - DIM1
T	DIM3 - DIM2
	DIM4 - DIM1
BOX	2*DIM4 - DIM1
	2*DIM3 - DIM2
CROSS	DIM4 - DIM3
H	DIM4 - DIM3
T1	DIM4 - DIM1
I1	DIM3 - DIM4
CHAN1	DIM3 - DIM4
Z	DIM3 - DIM4
CHAN2	DIM2 - DIM3
	2 * DIM1 - DIM4
bordered	DIM4 - DIM1
	DIM3 - DIM2
BOX1	DIM4 +DIM3 - DIM2
	DIM5 +DIM6 - DIM1
HEXA	2 * DIM1 - DIM2
HAT	2 * DIM2 - DIM1
	2 * DIM2 - DIM3
L	DIM3 - DIM2
	DIM4 - DIM1
HAT1	DIM3 - DIM1
	2* DIM4 - DIM2
	2* DIM4 +DIM5 - DMI2

Note that the L cross-section type is only available for the PBEAML.



If the user supplies cross-section types that are in addition to the MSC supplied types, constraints can be specified as a part of the specification (see [Adding Your Own Beam Cross-Section Library](#)). The constraints are applied once for each designed PBARL while the PBEAML has constraints for each designed section (possible sections are End A, End B and up to nine intermediate stations). The constraints are evaluated exactly as listed in [Table 2-6](#) with no normalization.

Gradient Evaluation

The gradient evaluation of [Figure 2-7](#) proceeds in 7 steps:

- The gradients of the properties are evaluated.
- Gradients of active constraints on type-1 responses are evaluated.
- Gradients of active constraints on type-2 responses are evaluated.
- Gradients of active constraints on type-3 responses are evaluated.
- Gradients of active constraints on dependent design variables are evaluated.
- Gradients of active property constraints are evaluated.
- Gradients of active beam library constraints are evaluated.

In addition, the gradient of the objective, which is one of the responses is always calculated.

Not all these steps need to be present in a particular design task. [Optimization](#) has provided a definition of active constraints; basically the concept is that the optimizer selects a reduced set of the retained constraints to deal with when it requires a gradient evaluation. The approximate model takes advantage of this by only computing the gradients of this reduced set. Each of the gradient steps is now described.

Gradients of the Properties

Gradients of all the properties are computed with respect to the design variables . This computation is a simple matter in the case of type-1 properties since they are a linear function of the design variable ([Equation \(2-2\)](#)) and the gradient is

$$\frac{\partial p_j^1}{\partial x_i} = C_{ji} \quad (2-140)$$

where the superscript 1 denotes a type-1 property.

Finite difference techniques are applied for the type-2 properties of [Equation \(2-5\)](#) with properties being perturbed one design variable at a time to provide

$$\frac{\partial p_j^2}{\partial x_i} = \frac{f(\mathbf{X}^0 + \Delta \mathbf{X}_i, C) - f(\mathbf{X}^0 - \Delta \mathbf{X}_i, C)}{2\Delta x_i} \quad (2-141)$$

where the $\mathbf{X}^0 + \Delta x_i$ notation is meant to imply that the ith design variable is perturbed while the remaining variables are held at their current values. The gradient of each type-2 property then is a vector with terms for each independent design variable.



Beam library properties (also called type-3) require chain-rule operation to compute a property derivative of the form

$$\frac{\partial p_j^3}{\partial x_i} = \sum_{k=1}^{ndim} \frac{\partial p_j^3}{\partial \text{DIM}_k} \frac{\partial \text{DIM}_k}{\partial x_i} \quad (2-142)$$

The second term in the chain rule is obtained from a type-1 gradient as given in [Equation \(2-140\)](#). The first term is the sensitivity of the property with respect to the dimension. For section types in MSC Beam Library, this sensitivity is provided analytically based on the equations for the beam properties. For example, the area of a tube cross-section is computed as

$$A = \pi(\text{DIM1}^2 - \text{DIM2}^2) \quad (2-143)$$

Then

$$\frac{\partial A}{\partial \text{DIM1}} = 2\pi\text{DIM1} \quad (2-144)$$

For user supplied cross sections, you have the option of supplying the analytic gradients or specifying that they are to be computed using central differencing techniques much along the lines of [Equation \(2-141\)](#).

Gradients of Type-1 Responses and of Active Constraints from Type-1 Responses

The gradients of all the type-1 responses are computed with respect to all the design variables in a two step process. In the first step, the direct sensitivities with respect to the design variables and the type-2 and type-3 properties are computed. The second step applies the approximation technique used in the function evaluation to obtain a gradient consistent with the predicted response

$$\frac{dr_j^1}{dx_i} = \frac{\partial r_j^1}{\partial x_i} + \sum_k \frac{\partial r_j^1}{\partial p_k^2} \frac{\partial p_k^2}{\partial x_i} + \sum_m \frac{\partial r_j^1}{\partial p_m^3} \frac{\partial p_m^3}{\partial x_i} \quad (2-145)$$

where the first term on the right is available directly from the sensitivity analysis of [Equation \(2-23\)](#), as are the

$$\frac{\partial r_j^1}{\partial p_k^2} \text{ and } \frac{\partial r_j^1}{\partial p_m^3}$$

terms. The

$$\frac{\partial p_k^2}{\partial x_i} \text{ and } \frac{\partial p_m^3}{\partial x_i}$$

terms are the property gradients just discussed in [Equation \(2-143\)](#) and [Equation \(2-144\)](#), respectively.

Given the gradient of the type-1 response, the calculation of the gradient of the constraint based on the response depends on the approximation used in calculating the response. For the direct approximation, the constraint gradient is given by

$$\frac{\partial g_k}{\partial x_i} = \frac{1}{\text{BND}_k} \frac{dr_j^1}{dx_i} \quad (2-146)$$



where, from [Equation \(2-10\)](#), lower bound constraints have $k = 2j - 1$ and $\text{BND}_k = -\text{GNORM}$ while upper bound constraints have $k = 2j$ and $\text{BND}_k = \text{GNORM}$.

For the reciprocal approximation, the gradient is

$$\frac{\partial g_k}{\partial x_i} = \frac{1}{\text{BND}_k} \frac{dr_j^1}{dx_i} \left(\frac{x_i}{x_i^0} \right)^2 \quad (2-147)$$

where the k subscript and BND_k have the same meaning as given above.

If convex linearization has been selected, the direct gradient of [Equation \(2-146\)](#) is used if the product of

$$\frac{x_i}{\text{BND}_k} \frac{\partial r_j}{\partial x_i} > 0.0$$

while the reciprocal approximation of [Equation \(2-147\)](#) if this product is less than zero.

Gradients of Type-2 Responses and of Active Constraints on Type-2 Responses

The gradient of a type-2 response is required when the response is the objective, when the response is constrained directly or when it is used in a type-3 or other type-2 response. The type-2 response can be a function of design variables, designed properties (both type-1 and type-2 properties), design responses (both type-1 and type-2 responses) and/or grid locations. With the exception of the grid locations and the type-2 responses, the gradients of these components have already been discussed. The grid location gradient can be obtained directly from the basis vector relationship of [Equation \(6-5\)](#). For the sake of this discussion, we can rewrite the type-2 response equation of [Equation \(2-7\)](#) into a condensed form of

$$r_j^2 = f(\mathbf{ARG}) \quad (2-148)$$

where ARG can be any argument in the response. A chain rule operation is then applied to obtain the overall gradient

$$\frac{dr_j^2}{\partial x_i} = \sum_k \left[\frac{f(\mathbf{ARG} + \Delta \mathbf{ARG}_k) - f(\mathbf{ARG} - \Delta \mathbf{ARG}_k)}{2 \Delta \mathbf{ARG}_k} \right] \frac{\partial \mathbf{ARG}_k}{\partial x_i} \quad (2-149)$$

The $\Delta \mathbf{ARG}_k$ is a finite difference change in a component and $\frac{\partial \mathbf{ARG}_k}{\partial x_i}$ is the gradient of the component with respect to the design variable and has already been calculated ($\frac{\partial \mathbf{ARG}_k}{\partial x_i}$ is 1.0 when \mathbf{ARG}_k is x_i).

The $\Delta \mathbf{ARG}_k$ term is calculated using

$$\Delta \mathbf{ARG}_k = \begin{cases} FDCH \cdot |\mathbf{ARG}_k| & (FDCH \cdot |\mathbf{ARG}_k| > FDCHM) \\ FDCHM & \text{otherwise} \end{cases} \quad (2-150)$$

FDCH and FDCHM are 0.001 and 1.0^{-6} , respectively.

The gradients of the constraints based on type-2 two responses are simply



$$\frac{\partial g_k}{\partial x_i} = \frac{1}{BND_k} \frac{dr_j^2}{dx_i} \quad (2-151)$$

where the k subscript and the BND_k value have the same meaning as they do for the type-1 response of [Equation \(2-146\)](#).

Gradients of Active Type-3 Responses

The gradient of a type-3 response is required when the response is the objective or when the response is constrained and the constraint is active. The type-3 response can be a function of design variables, designed properties (both type-1 and type-2 properties), design responses (both type-1 and type-2 responses) and/or grid locations. The user has a number of options for calculating these gradients and this is governed by the GRDTYP specified in the R3SGRT subroutine introduced in “Creating a DRESP3 Entry”. Options are:

1. GRDTYP=1 - Analytical gradients are supplied by the user. These gradients are a function of the approximate optimization task.
2. GRDTYP=2 - Analytical gradients are supplied by the user. These gradients are taken to be invariant during the approximate optimization task.
3. GRDTYP=3 - Finite difference gradients are computed and they are a function of the approximate optimization task.
4. GRDTYP=4 - Finite difference gradients are computed before the approximate optimization task and are considered invariant during the task.

The same chain rule formulation as that given in [Equation \(2-149\)](#) for type-2 gradients is applicable for type-3 gradients when the finite difference method is applied. If the analytical gradient method is used, the user needs to provide the gradient information via the R3SVALD subroutine. Gradients of the constraints based on type-3 responses follow the method shown in [Equation \(2-151\)](#).

Gradients of Active Dependent Design Variables

The gradient of an active constraint imposed on the j -th dependent design variable with respect to the i -th independent variable is given by combining [Equation \(2-1\)](#) with [Equation \(2-139\)](#) and differentiating to get

$$\begin{aligned} \frac{dg_L^{DDV}}{dx_i} &= \frac{-c_i}{|XLB|} \\ \frac{dg_u^{DDV}}{\partial_i} &= \frac{c_i}{|XUB|} \end{aligned} \quad (2-152)$$

Gradients of Active Property Constraints

The gradient of an active constraint imposed on the j -th designed property with respect to the i -th design variable is given by



$$\begin{aligned}\frac{dg_L^P}{dx_i} &= \frac{-1}{|p_j^L|} \frac{\partial p_j(x)}{\partial x_i} \\ \frac{\partial g_u^P}{\partial x_i} &= \frac{1}{|p_u^L|} \frac{\partial p_j(x)}{\partial x_i}\end{aligned}\quad (2-153)$$

where the property gradients have been defined in [Equation \(2-140\)](#), [Equation \(2-141\)](#), and [Equation \(2-142\)](#).

Gradients of Active Beam Library Constraints

From [Table 2-6](#), it is seen that the gradient of the j -th beam library constraint with respect to the i -th design variable is given by

$$\frac{\partial g^{BL}}{\partial x_i} = \frac{\partial g^{BL}}{\partial DIM_j} \frac{\partial DIM_j}{\partial x_i} \quad (2-154)$$

This is a straight-forward calculation. As an example, suppose a CHAN section type is being designed and the DIM4 dimension is defined as being equal to $0.1 \cdot x_5$. Then the gradient of the first CHAN constraint of [Table 2-6](#) with respect to x_5 is simply

$$\frac{\partial g_{CH1}^{BL}}{\partial x_5} = 2.0 \cdot 0.1 = 2.0 \quad (2-155)$$



Tests for Convergence

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, lower level is at the optimizer level; the second, 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 discussed in [The IPOPT Algorithm](#).

Convergence of Design Cycles: Hard and Soft Convergence

Figure 2-1 has indicated that a convergence check is made immediately following the structural analysis. This is a somewhat simplified representation in that, in reality, two methods are used to test for convergence with respect to overall design cycles. These methods are denoted as soft convergence and hard convergence as shown in the modified version of **Figure 2-1** shown in **Figure 2-15** which highlights the convergence aspects of the design cycle loop. Soft convergence is based on the results of the approximate optimization, while hard convergence is based on finite element analysis results.

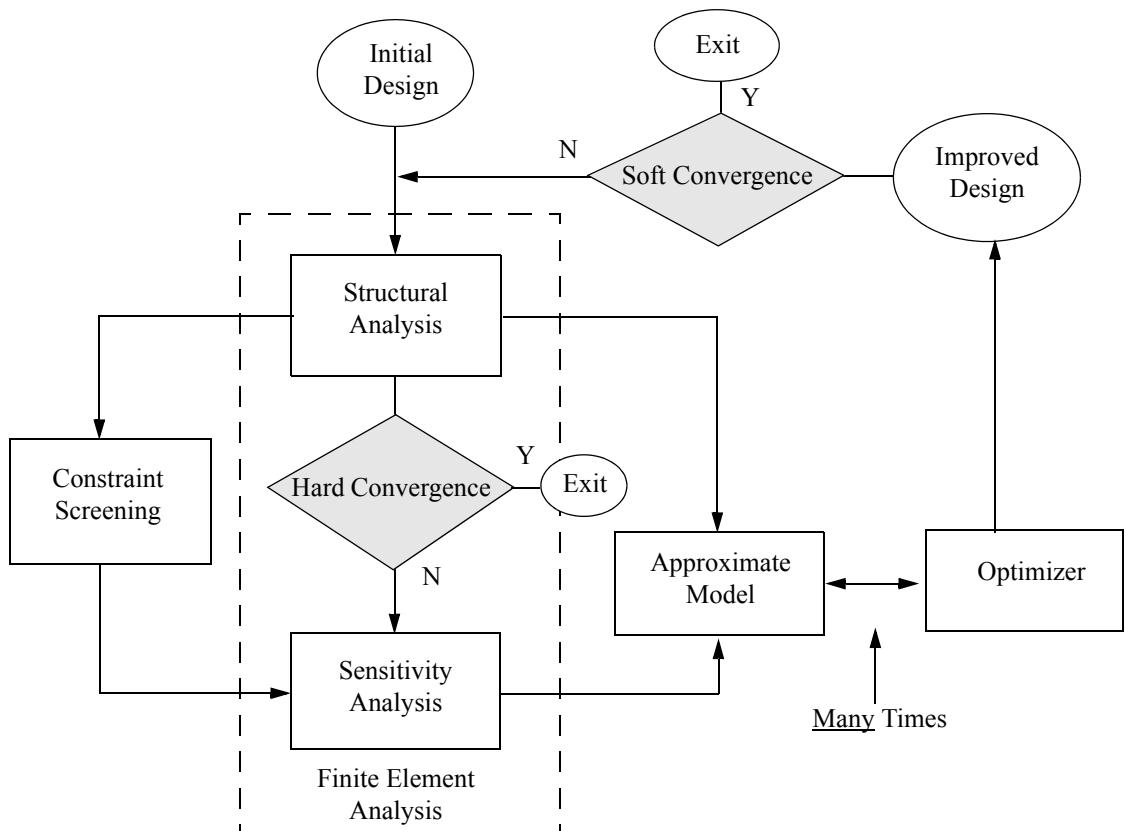


Figure 2-15 The Design Cycle Loop with Emphasis on the Convergence Tests



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.

Soft convergence compares the design variables and properties output from the approximate optimization with those of the input to the approximate optimization. If these variables/properties have not changed appreciably, another finite element analysis may not be warranted. For example, if a particular analysis model has a high solution cost and the optimizer has not changed the design to any significant 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. Soft convergence is always tested, but, by default, does not result in program termination. Setting PARAM SOFTEXIT to YES will result in program termination following a determination that soft convergence has been achieved.

Once a corresponding approximate optimum is found, the finite element model, in terms of the design properties and grid locations, is updated. 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 analysis) 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 variables and properties output from the approximate optimization with those of the input to the approximate optimization. If these variables/properties have not changed appreciably, another finite element analysis may not be warranted. For example, if a particular analysis model has a high solution cost and the optimizer has not changed the design to any significant 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. Soft convergence is always tested, but, by default, does not result in program termination. Setting PARAM SOFTEXIT to YES will result in program termination following a determination that soft convergence has been achieved.

On the initial cycle of the Design Cycle loop shown in [Figure 2-15](#), 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 this design with the prior design.

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.

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 is met. This default maximum number of iterations is 5 but can be changed on the DOPTPRM entry using DESMAX.



The following paragraphs discuss soft and hard convergence decision logic.

Soft Convergence Decision Logic

A schematic of the soft convergence decision logic is shown in [Figure 2-16](#). The result of the test is a true or false value for the logical variable SOFTCV. The design cycle terminates only if SOFTCV is ‘TRUE’ and the parameter SOFTEXIT is ‘YES’.

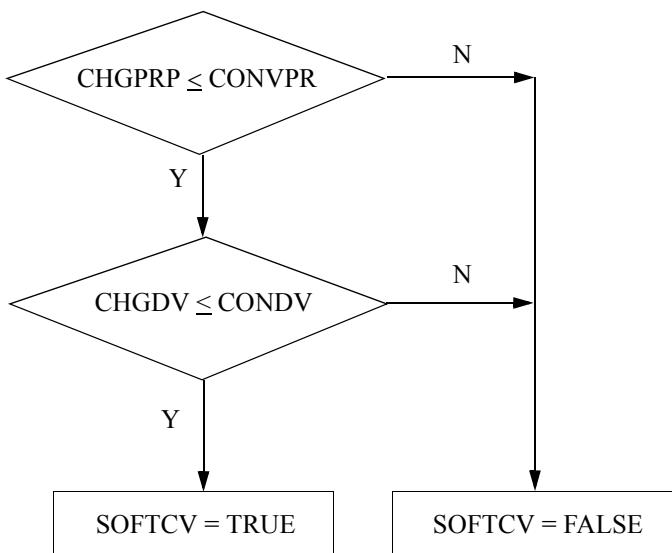


Figure 2-16 Soft Convergence Decision Logic

From [Figure 2-16](#), soft convergence is based solely on the relative changes in the properties and design variables. The definitions of the parameters in the figure appear in [Table 2-7](#) along with their default values. These values may be modified using the DOPTPRM entry.



Table 2-7 Convergence Criteria Parameters (the P and P-1 Superscripts Refer to the Current and Previous Design Cycle)

Internal Variable	Definition	Parameters	Default
CHGOBJ	$\left \frac{OBJ^{(P)} - OBJ^{(P-1)}}{OBJ^{(P-1)}} \right $	CONV1	0.001 (1.0E-5 for topology optimization)
ACHOBJ	$OBJ^{(P)} - OBJ^{(P-1)}$	CONV2	1.0E-20
CHGPRP	$\max_{1 \leq i \leq NPROP} \left(\left \frac{P_i^{(P)} - P_i^{(P-1)}}{P_i^{(P-1)}} \right \right)$	CONVPR	0.001
CHGDV	$\max_{1 \leq i \leq NDV} \left(\left \frac{x_i^{(P)} - x_i^{(P-1)}}{x_i^{(P-1)}} \right \right)$	CONVDV	0.001 (1.0E-4 for topology optimization)
CONMAX	$\max_k \{g_k(x)\}$	GMAX	0.005

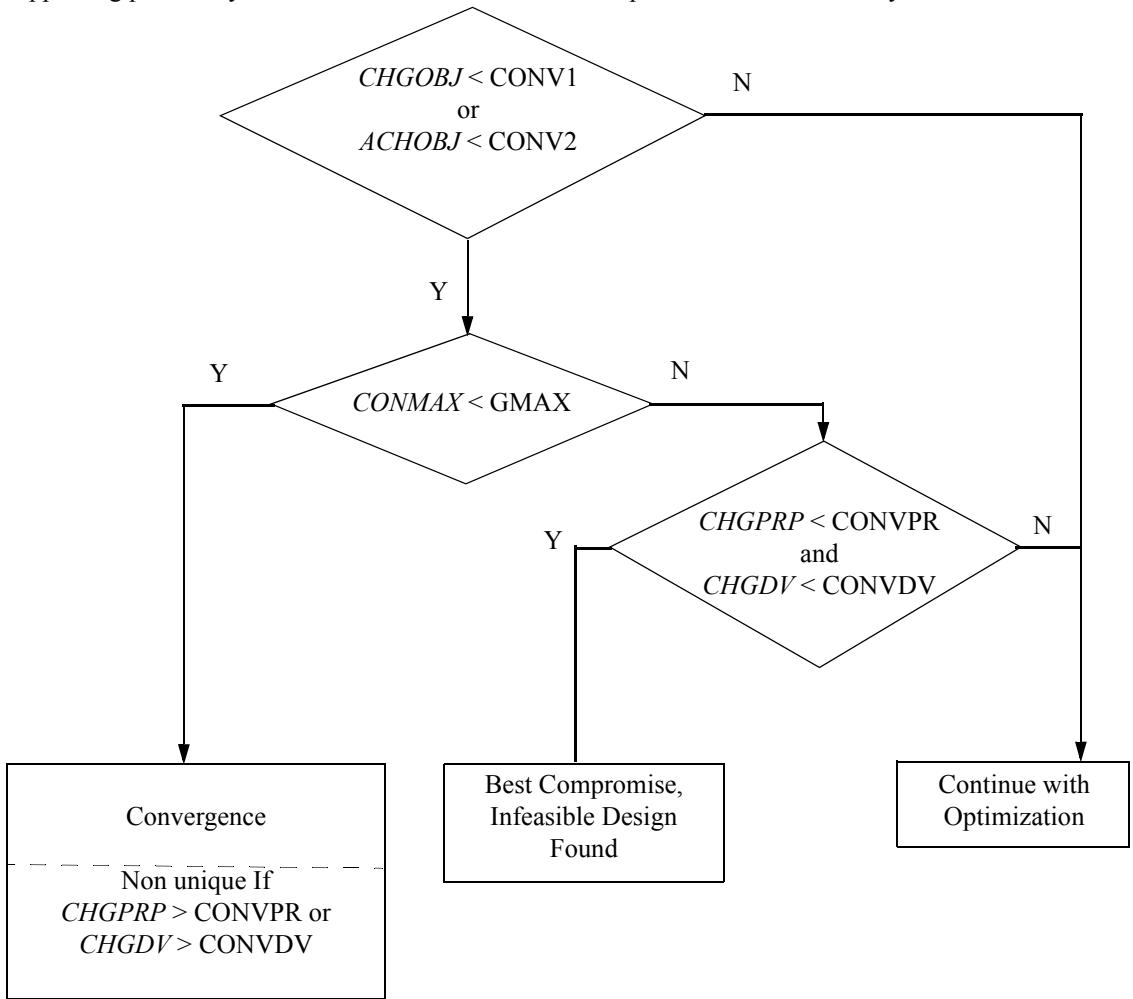
The assumption can be made that if the changes in the design variables and properties are not appreciable, the objective and the constraints are unchanged and there is nothing to be gained from continuing.

Hard Convergence Decision Logic

Hard convergence is used to decide whether or not the design cycle process should continue, providing that the maximum allowable number of design cycles (DESMAX) has not yet been reached. The satisfaction of hard convergence is always sufficient to stop the design process.



A flowchart of these criteria is shown in [Figure 2-17](#) with the corresponding definitions of the terms appearing previously in [Table 2-7](#). Note that one of three possible conclusions may be reached.



[Figure 2-17](#) Hard Convergence Decision Logic

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 a kilogram 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.

Special Convergence in Topology Optimization

Topology optimization can converge using criteria that are somewhat different from that given above. In this case, it is still necessary that the current and previous design have no violated constraints but somewhat relaxed criteria are imposed once this condition is met. Hard convergence is now achieved when:

$\text{CHJOB} < 0.005$

And

The percentage of grey elements is $< 30. * \text{XINIT}$

Where a grey element is one whose value is $0.3 < x_i < 0.7$. XINIT is set by the rules given on the TOPVAR entry and it is seen that if it is 0.3, the percentage of grey elements must be less than 9%.

Non-Unique Optima

Even if hard convergence has been achieved, the user should check to see if the relative changes in properties CHGPRP or the relative changes in design variables CHGDV are satisfied. If they are, then the design process has converged to a unique design. If not, a non unique design may have been found.

From a design perspective, convergence to a non unique design is extremely useful information. This would indicate that the optimum properties could be modified by the engineer with correspondingly little change in the objective function and/or constraints. (In other words, the optimal design exhibits low sensitivity with respect to changes in the design variables.) A design space of this type is shown in [Figure 2-18](#) 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.



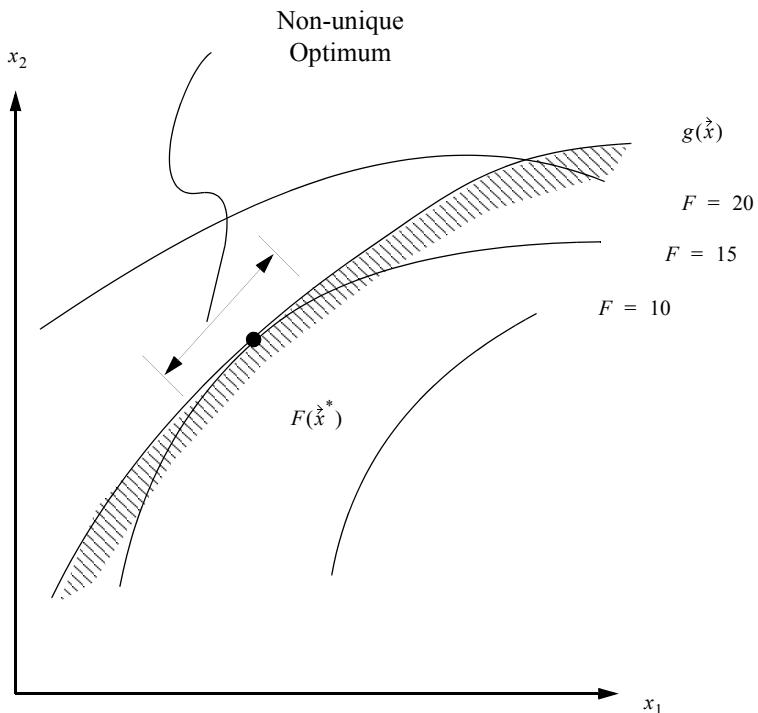


Figure 2-18 Non-unique Optimum

Compromise Infeasible Design

Returning to the hard convergence decision logic of [Figure 2-17](#), 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, optimization can continue 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 2-19](#) where the optimal design is that which minimizes the sum of the constraint violation.



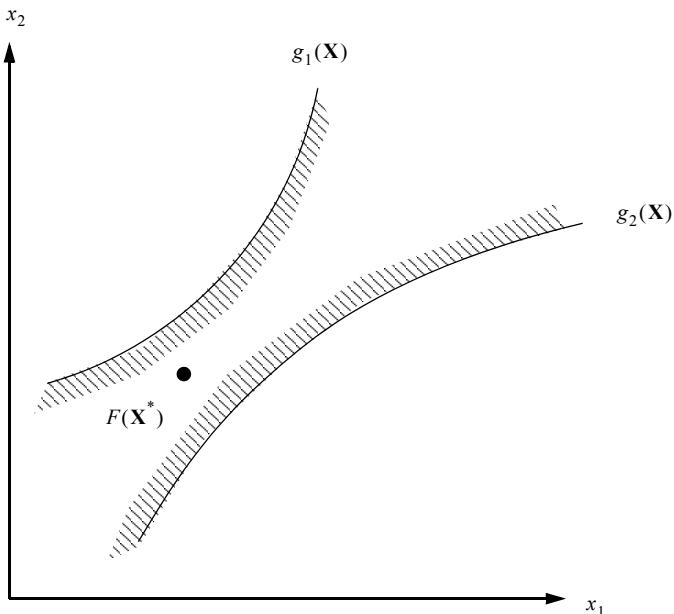


Figure 2-19 Design Space, No Feasible Solutions

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. A discussion of the output from both the soft and hard convergence checks, given in “[Design Optimization Output](#)” starting on [page 269](#), is particularly relevant to the stopping criteria in that it contains a list of all the possible messages that can occur at the end of an optimization task.





3

Developing the Design Model for Sensitivity and Optimization

- Overview of Design Modeling 109
- Case Control for Design Optimization 111
- Defining the Design Variables 113
- Relating Design Variables to Properties 116
- Designating the Design Responses 119
- Defining the Objective Function 127
- Defining the Constraints 129



The previous chapter provided an in-depth description of how Design Sensitivity and Optimization has been implemented within the MSC Nastran product. A key concept in that description was that of the design model ([The Design Model](#)). 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 judgment if the design results are to be of any practical use.

The essential features of the design model are that it 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).
- Identify and/or develop responses that are to be used in the 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, with reference to the text user interface that is used to define the design model. In connection with this chapter, you will want to refer to [Input Data](#) that explains the text interface in detail and [Example Problems](#).

The examples in Chapter 7 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 MSC Nastran.



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:



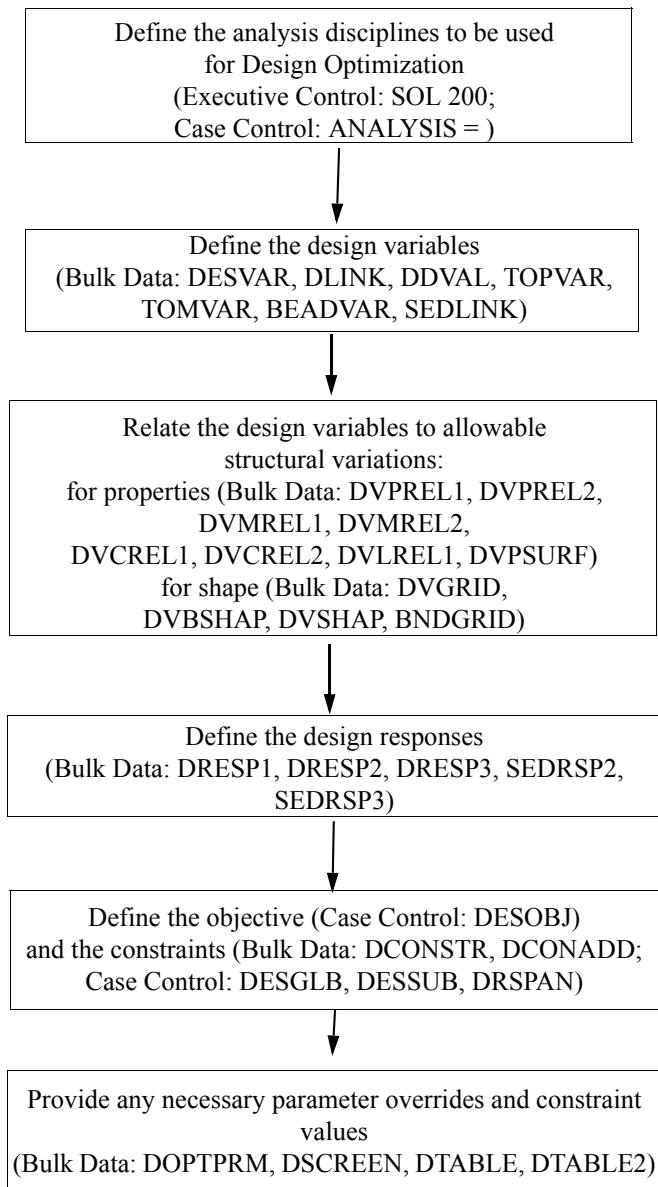


Figure 3-1 Design Modeling Process

The following sections discuss each of these items in greater detail.



Case Control for Design Optimization

Case Control commands in MSC Nastran provide the overall direction for the analysis to be carried out as well as the data recovery operations to be performed. In this context, a powerful feature of design sensitivity and optimization in MSC 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.

As an 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 can quite easily introduce these different analysis types in Solution 200 (the solution sequence for design optimization) with the following Case Control packet:

```
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
    tstep = 10
begin bulk
```

Note some of the features in this example:

SOL 200	This is an Executive command indicating Solution 200, the solution sequence for design sensitivity and optimization, is to be invoked.
ANALYSIS	This Case Control command indicates 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
DCEIG	Direct complex eigenanalysis
DFREQ	Direct Frequency
MCEIG	Modal complex eigneanalysis
MFREQ	Modal Frequency
MTRAN	Modal Transient
SAERO	Steady Aeroelastic
FLUTTER	Flutter
DESOBJ	Selects the design objective from a response defined in Bulk Data with DRESP1, DRESP2 or DRESP3.
DESSUB	Selects constraints to be applied at a particular subcase level from a DCONSTR or DCONADD set defined in Bulk Data.

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
```

In general, case control requests are for data recovery only; and 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 [Designating the Design Responses](#)). You do not need to explicitly request this data recovery unless you want to view the results. There are some special cases however that you need to know about where output options do play a role in specifying design responses:

- 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 STRAIN example, STRAIN(STRCUR) representation is the default. If STRAIN(FIBER) is provided, data recovery and design optimization responses are both computed using this form. Two other instances like this are the VONMISES vs. MAXS option on the STRESS/STRAIN commands and the (REAL or IMAG) vs. PHASE option of many case control commands in a Dynamic Response analysis. Defaults are listed in the *MSC Nastran Quick Reference Guide*.
- Shape optimization and superelement optimization each have some additional Case Control considerations beyond those related to data recovery. [Design Variables in Superelement Design Modeling](#) discusses the Case Control structure for superelement design models. In addition, [Example Problems](#) includes examples of both, shape and superelement optimization.



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.
- Add additional terms to the design task that are not explicitly related to the finite element analysis.

This third category allows you to stretch your imagination when applying Design Optimization in MSC Nastran. The availability of type-2 and type-3 responses allows you to consider design conditions that are not directly related to the finite element analysis. Although you probably do not want to optimize text book mathematical programming tasks using MSC Nastran, it is not outside the range of possibility. On a more realistic note, the example of [Acoustic Optimization](#) uses an added design variable in a novel way to minimize the maximum acoustic pressure in the analysis.

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.

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, bounds (or side constraints), an optional move limit for the approximate optimization and, for discrete variable optimization, a pointer to a list of discrete values the design variable can assume. The bounds limit the region of search with respect to each variable as

$$x_i^L \leq x_i \leq x_i^U \quad (3-1)$$

The optimizer will never propose a design that violates these bounds.

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, DDVAL
$
DESVAR, 10, AREA1, 0.05, 0.01, 0.1, 0.2
DESVAR, 11, THICK1, 0.03, 0.01, 0.08
$
```

The label fields are strictly to help you in characterizing the variable and you have complete freedom in the name you specify for the variable. In this case, 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

$$\begin{aligned} 0.01 \leq x_{10} &\leq 0.1 \\ 0.01 \leq x_{11} &\leq 0.08 \end{aligned} \quad (3-2)$$

The DELXV field, which is 0.2 for the AREA1, limits the amount this design variable can change during an optimization cycle by 20%. The THICK1 variable has this field blank so the move limit for this second variable is the value of the DELX parameter you can provide as a default for all of the variables on the DOPTPRM entry. If you don't specify the DELX parameter either, the default move limit is 50%.

There is one more optional field on the DESVAR entry that enables you to perform a discrete optimization task as explained in [Special Topics](#). This is the DDVAL attribute on the DESVAR entry and the integer value points to a DDVAL Bulk Data entry with this ID. The presence of this integer triggers discrete variable optimization that produces a design with design variable values chosen from the discrete set of real numbers given on the DDVAL entry.

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

$$\mathbf{X} = [x_1, x_2, \dots, x_n]^T \quad (3-3)$$

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.

A question that comes up frequently is: How many design variables can MSC Nastran handle in a design task? There is no single answer for this question. The MSC Nastran code is written to avoid fixed limits and the design sensitivity and optimization capability is no exception to this. Further, the answer depends on the computer resources you have available, your tolerance for expending these resources and your time waiting for the results and also on the type of problem you are trying to solve. Given all that, it is still useful to provide a guideline. Problems with up to a thousand design variables are considered routine in MSC Nastran while problems that have several thousand variables can be considered extreme. In cases where the design task is "easy," (i.e., find the set of design variables that minimizes the weight while satisfying a single displacement limit), it's possible to consider upwards of 10,000 variables.

The above paragraph refers to the MSCADS algorithms that are included as part of the design optimization module. If the design task has several thousand design variables, we recommend that the IPOPT algorithm be used. The main purpose for this IPOPT is to perform topology optimization, however, the IPOPT optimization algorithm can be used for other design tasks with up to one million design variables. Note that is not necessary for you to select the optimization algorithm. If a preference has not been specified, MSC Nastran will select the most appropriate method based on the number of design variables and the number of active constraints.

Design Variables and the Finite Element Model

Recall that the problem we are trying to solve is defined by the basic optimization problem statement of [Equation \(1-1\)](#) through [Equation \(1-5\)](#). 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

$$M = M(\mathbf{P}, \mathbf{G}) \quad (3-4)$$

where the vector \mathbf{P} is the collection of properties that describe the model and the vector \mathbf{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

$$\begin{aligned} \mathbf{P} &= \mathbf{P}(\mathbf{X}) \\ \mathbf{G} &= \mathbf{G}(\mathbf{X}) \end{aligned} \quad (3-5)$$

By direct substitution of (3-5) into (3-4),

$$M = M(\mathbf{P}(\mathbf{X}), \mathbf{G}(\mathbf{X})) \quad (3-6)$$

or simply,

$$M = M(\mathbf{X}) \quad (3-7)$$

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.

Optimization Design Variables

The discussion in this section has been terms of the DESVAR Bulk Data entry that is used in standard design optimization. For topology optimization, there typically is a design variable for every finite element and it would be impractical to ask the user to define each of these individually. Instead, a TOPVAR Bulk Data entry is used as described in [TOPVAR](#). This entry identifies a property ID that can be associated with many finite elements. In this case, design variables are created internally for each of these elements. The initial values and upper and lower bounds on each of these design variables are obtained from the TOPVAR entry. The density and the Young's Modulus of the element are then governed by the relationships shown in [Equation \(7-1\)](#). Similarly, for topometry optimization the TOMVAR entry spawns a design variable for each element associated with the identified property ID. For topography and the BEADVAR entry, a separate design variable is spawned for each element associated with the identified property ID to create a shape variable.



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 for sizing optimization. [Relating Design Variables to Shape Changes](#) discusses shape variations for shape optimization.

Analysis model properties are defined as those quantities which appear as real numbers on Bulk Data property entries. As discussed in [Overview of Fundamentals](#), four different types of properties are supported for MSC Nastran design sensitivity and optimization: element properties, material properties, connectivity properties and load properties. Element properties are items such as plate thicknesses, area moments of inertia and elastic spring stiffnesses while material properties are, for example, Young's modulus and Poisson's ratio. Connectivity properties are on element connectivity entries and include such items as beam offsets and concentrated masses. Load properties are currently limited to real numbers appearing on FORCE, MOMENT and LOAD entries.

You can relate design variables to analysis model properties in either of two ways: with linear relations or with MSC Nastran's equation input capability. A linear relation is described using one of the DVxREL1 (Design Variable-to-Property RELation, type-1) Bulk Data entry, where the x is replaced with a 'P' for element properties, an 'M' for material properties, a 'C' for connectivity properties and a 'L' for load properties.

As a review from [The Design Model](#), linear design variable-to-property relations are of the form

$$p_j = C_o + \sum_i C_i x_i \quad (3-8)$$

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

As described as part of the DVxREL1 descriptions (see [DVPREL1](#)), a special case is provided for the case when C_0 and C_i ($i > 1$) are 0.0 and the user wishes to use the actual property value for C_1 . In this case, if the term PVAL is used as input for C_1 , the actual property value is used as the coefficient.

An equation, or type-2 relation, is defined using a DVxREL2 Bulk Data entry which references a DEQATN (Design EQuATioN) entry. A property expressed using these relations can be written as

$$p_j = f(\mathbf{X}, \mathbf{C}) \quad (3-9)$$

where the j -th property is a function of a collection of design variables and constants. The function is defined using a DEQATN Bulk Data entry. Note that there is no support for a type-2 load property.

The same design variables may appear in either type-1 or type-2 relations. The block diagram of [Figure 3-2](#) may be helpful in understanding the role of the DVPREL1 and DVPREL2 entries in defining property variations with the DVCREL1, DVMREL1, DVCREL2, and DVMREL2 entries having a similar structure.



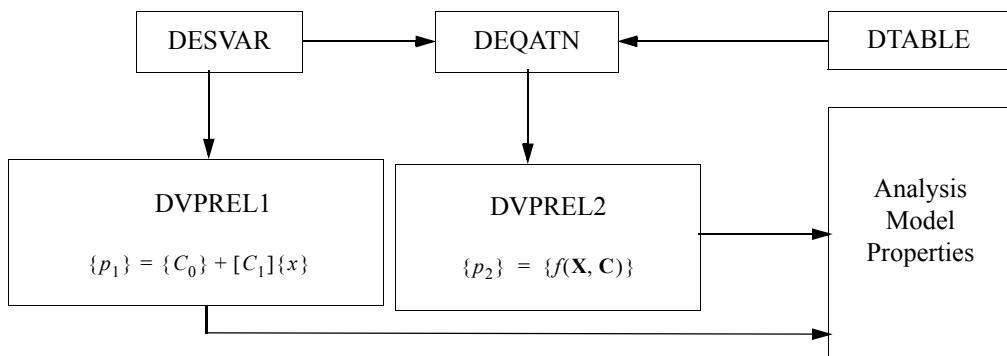


Figure 3-2 Design Variable-to-Property Relations

The DVPREL1 entry defines linear property variations and requires only design variable input. All necessary constants are supplied directly on the entry. On the other hand, the DVPREL2 entry relies on an equation to describe the (generally) nonlinear property variations. This equation may have design variables (DESVAR) as well as table constants (DTABLE, DTABLE2) as input. As shown in [Figure 3-2](#), the result is again an analysis model property variation.

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 3-3](#). The property IDs MID1 through MIDm are referenced on the DVMREL1 and DVMREL2 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 DVMREL-type entries. For example, to modify the material density, all that is required is to reference a MAT1 ID on a DVMREL entry. All elements that use that material will then change accordingly.

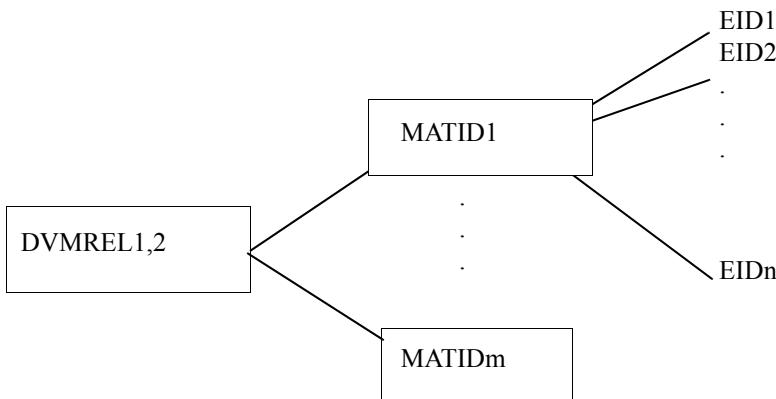


Figure 3-3 Design Model Reference by Property Entry

You must identify a selected property entry item by its name on the property Bulk Data Entry, where the name is given in the description of the entry in the ***MSC Nastran Quick Reference Guide***. For example, the area of a beam on a PBAR has the name **A** while the thermal expansion coefficient on a MAT1 entry also has the name **A**. [Table 2-1](#) through [Table 2-4](#) list the available property names.

To illustrate both type-1 and type-2 design properties consider the following hypothetical example that calls for designing a plate under the assumption that a change in the thickness of the plate causes a change in a concentrated mass that is used to estimate such things as paint wiring and rivets. Specifically suppose the relation between the mass and stiffness is given as:

$$\text{mass} = a_0 + a_1(t)^{1/2}$$

and a_0 and a_1 are 0.1 and 0.5 respectively. The analysis model and the design model can be written as:

```

PSHELL 1      1      0.15    1
CONM2 10100   10102   1.0
DESVAR 1      T-PLATE 1.0   0.001  10.0
DVPREL1 1     PSHELL  1      T      0.01
           1      1.0

DVCREL2 1     CONM2    10100   M          40
DESVAR 1
DTABLE A0      A1
DTABLE A0      0.1      A1      0.5
DEQATN 40     MASS(X,A0,A1) = A0 + A1 * SQRT(X)

```

It is not necessary to define constants using the DTABLE entry, but this can enhance the readability of the DEQATN when it is used in a large model. Also note that it is not necessary for the value of the designed property be the same as the corresponding analysis value. In the above example, the CONM2 has an M value of 0.35 while the designed M value is

$$0.1 + .5(1.0)^{1/2} = 0.6$$

The designed property value always takes precedence over the analysis property.

See [Special Considerations When Designing One-Dimensional Bending Elements](#) for additional comments on this topic.



Designating the Design Responses

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

DRESP1 Bulk Data Entries

DRESP1 Bulk Data entries define type-1, or first-level responses. These responses are available directly from an MSC Nastran analysis. Structural weight, displacements at grid points, element stresses, and so on, are all examples of type-1 responses. See the DRESP1 Bulk Data entry description for a list of available response types in design optimization. The DRESP1 entries all require input of a response ID, response label, and response type (e.g., WEIGHT) but the remaining data input is response type dependent and this entry is therefore discussed at length in [DRESP1](#).

DRESP2 Bulk Data Entries

DRESP2 Bulk Data entries define type-2, or second-level responses. This class of responses is also referred to as “user-defined” or “synthetic” since the DRESP2 utilizes the equation input feature in MSC Nastran in a fashion similar to the of the user defined properties (DVPREL2,DVMREL2 and DVCREL2) discussed in [Relating Design Variables to Properties](#). With the DRESP2 entry and its companion DEQATN entry (Design EQuATioN), responses such as error functions, stress averages, and so on can be defined. First-level responses, design variables, grid coordinate values, table constants, designed properties, and even other second-level responses may all be used as input to these user-defined responses. [Figure 3-4](#) is a diagram of the DRESP1 and DRESP2 data structure.



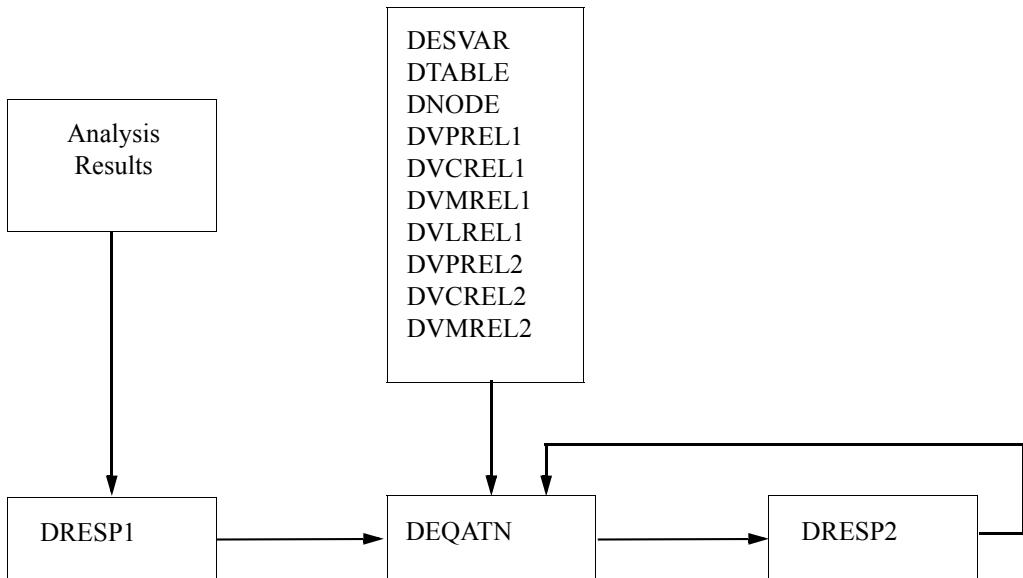


Figure 3-4 Second-Level Responses

The DRESP2 entry has a number of features you need to understand to make the most effective use of this powerful tool. [DRESP2](#) provides a more in-depth discussion, but it is noted for now that one option available on the DRESP2 entry is to invoke a standard type of operation that does not require a DEQATN. For example, the SSQ option automatically performs a sum of squares operation on the DRESP2 arguments.

$$\text{SSQ} = \sum_{i=1}^{\text{NTERMS}} (\text{XITEM})_i^2$$

Therefore, you do not need to develop a DEQATN that performs this routine, but potentially lengthy, summation. Other available standard options are discussed in [Table 4-4](#).

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, ATT A,   ATT B,   ATT C,   +
$+,      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 510      U( UX, UY ) = SQRT( UX**2 + UY**2 )
$... SYNTHETIC RESPONSE:
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 arguments are the two first-level responses DRESP1 501 and 502. DRESP2 520 can now be used as the objective function or invoked by a constraint.

A standard function can also be used in this case so that the input would be simply:

```
$... SYNTHETIC RESPONSE
DRESP2 520      U      RSS
DRESP1 501      502
```

Efficient Definition of Responses

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. A THRU option is not supported, however.

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 (the axial response is selected using a stress item code as explained in [STRESS](#)) 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 3-5](#) 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. When it is further considered that this constraint may be selected by multiple subcases, it is seen that just a pair of entries can possibly define thousands of stress constraints.

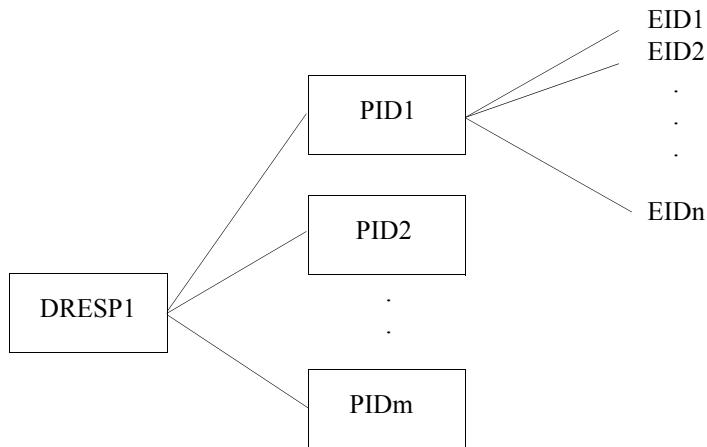


Figure 3-5 Identification of Element-Level Responses

These efficiencies in design response identification extend to type-2 and type-3 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:

```

$ Input to equation to compute max shears:
DRESP2, 201,   MAXS,   AVG,   '   ,   '   ,   '   ,   '   ,
+,      DRESP1, 101,   102
  
```

This single DRESP2 entry defines a maximum shear for every element in the group. Since the underlying first-level responses are element-level, the DRESP2 is as well.



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

1. 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.
2. A DRESP2 entry cannot self-reference the same DRESP2.
3. A DRESP2 can reference DRESP1 entries from separate subcases, but this requires special handling that involves the DRSPAN case control command.
4. If there is a desire to do parts superelements, the SEDRSP2 and SEDRSP3 entries must be used.

Example Using DRESP1 and DRESP2 Entries

Suppose we would like to include Euler buckling constraints in the design model for a simple, pin-ended rod elements. A representative element is shown in [Figure 3-6](#).

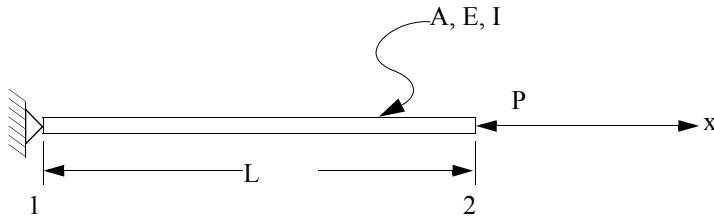


Figure 3-6 Pin-ended Rod Element

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

$$P_{CR} = \frac{\pi^2 EI}{L^2} \quad (3-10)$$

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

$$\left| \frac{P}{P_{CR}} \right| \leq 1 \quad (3-11)$$

which can be rewritten as

$$\frac{-PL^2}{\pi^2 EI} \leq 1 \quad (3-12)$$

Note that the critical load is a function of the area moment of inertia, yet this quantity is not a part of the analysis model specification for the ROD element. (Since the inertia of the ROD is not necessary to



compute the axial force, the inertia is not needed to perform the analysis.) If the cross-sectional area of the element is the design variable and a solid circular section is assumed, then

$$\begin{aligned} A &= \pi r^2 \\ I = \frac{\pi r^4}{4} &\Rightarrow I = \frac{A^2}{4\pi} \end{aligned} \quad (3-13)$$

and from [Equation \(3-12\)](#)

$$\frac{-4P(L)}{\pi E} \left(\frac{L}{A} \right)^2 \leq 1 \quad (3-14)$$

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.
GRID      2          10.   0.    0.
CROD     100       200    1     2
PROD     200       201    0.5
$ DESIGN MODEL
DESVAR   1          A200   0.5   0.25   0.75
DVPRELL1 301       PROD   200    A
           1          1.0
DRESP1   410       SA     FORCE   ELEM           2          100
DRESP2   501       EUL1   600
DESVAR   1
DTABLE   L         10.0    E     10.0E7
DRESP1   401
DTABLE   L         10.
DEQATN   600       EUL1(A,L,E,P) = -4. * P* L**2/(PI(1) * E * A**2)
$ note the use of PI function (see DEQATN entry description)
DCONSTR  500       501    1.0

```

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.

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 and a 2 in field 7 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 item codes. (See [Item Codes](#) in the *MSC Nastran Quick Reference Guide*.)

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. The order of the DRESP2 continuation lines is not interchangeable, with the order provided by the DRESP2 Bulk Data entry description.

This example lends itself to illustrating another feature of the DRESP2 entry: the ability to reference another DRESP2 entry. The example has been presented in terms of a single ROD entry and a single buckling constraint. In reality, it is likely that you will want to impose this type of constraint at many points in the structure and this could be done by duplicating the design model of the bulk data fragment for each constraint. This entails determining the length of each rod and using it in the equation. Since the form of the length calculation is shared across all the elements, it may be desirable to formulate a



DRESP2/DEQATN to just determine the length and to then use this in the buckling constraint. The bulk data fragment given above could then be redone as:

```

grid#      1          0.    0.    0.
grid#      2          10.   0.    0.
CROD     100       200    1      2
PROD     200       201    0.5
$ DESIGN MODEL
DESVAR   1        A200   0.5   0.25   0.75
DVPREL1  301      PROD   200    A
           1        1.0
DRESP1   410      SA     FORCE   ELEM      2          100
DRESP2   411      LENGTH 400
DNODE    1        1      1      2      1      3      2
           1        2      2      2      3
DEQATN   400      LENGTH(X1,Y1,Z1,X2,Y2,Z2) = SQRT((X2 - X1)**2 +
           (Y2 - Y1)**2 + (Z2 - Z1)**2 )
DRESP2   501      EUL1   600
DESVAR   1
DTABLE   E
DRESP1   410
DRESP2   411
DTABLE   E      1.0E7
DEQATN   600      EUL1(A,E,P,L) = -4. * P * L**2/(PI(1) * E * A**2)
DCONSTR 500      501      1.0

```

DRESP3 Bulk Data Entries

DRESP3 Bulk Data entries define type-3 or external responses. This bears some similarity to the DRESP2 just described with the important difference that the DRESP3 response is evaluated by invoking an external (to MSC Nastran) process via an application programming interface (API). Therefore, instead of invoking an equation for DRESP2, the DRESP3 entry identifies a GROUP or TYPE for the external response. The arguments for the DRESP3 are similar to DRESP2 so that the data structure diagrammed in [Figure 3-4](#) becomes:



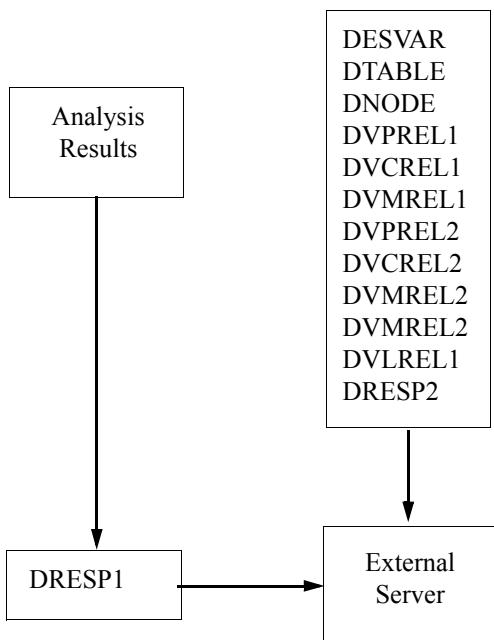


Figure 3-7 Third-Level Responses

Note that, unlike the DRESP2, a DRESP3 cannot reference another DRESP3. Also, there is an additional USRDATA argument than can be used to pass character data such as names of data files or options to take within the external server.

Since an external response is computed using a user-built server executable, defining an external response requires additional user actions to establish the association between the MSC Nastran program and the response server programs. Five user actions are required in setting up the external response in a design task:

1. Create a DRESP3 entry to define the response.
2. Write Fortran or C routines and build a server executable based on the server template routines.
3. Create a CONNECT entry to define an external response group.
4. Create a evaluator connection file to associate the external response group with the server program.
5. Submit the MSC Nastran job with the gmconn keyword that references to the evaluator connection file.

Descriptions of each user actions are given in [External Response](#), with much of the description similar to that found in [Building and Using the Sample Programs](#) in Chapter 7 of the *MSC Nastran Installation and Operations Guide*. An example application is shown in [External Response to Include Alternative Buckling Response](#).



Defining the Objective Function

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

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-2 kg or less and 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. There are three alternative remedies for this, depending on how other design variables affect the objective. If none of the design variables have a significant effect, you may not have formulated the best objective. In a weight minimization, this can occur because the calculated weight includes all the weight in the structure, including fixed weights that are not functions of the design variables. If these fixed weights dominate the overall weight, you can use a DRESP2 to subtract out this fixed weight so that the objective becomes a stronger function of the design variables. In our example, if 998 of the 1,000 kilograms are fixed, subtracting this number as shown in the following bulk data fragment will result in an objective that the optimizer will have an easier time in modifying.

```
TITLE = FOCUSED WEIGHT RESPONSE
DESOBJ = 100
.
.
.
BEGIN BULK
.
.
.
DRESP1    10      ENTWGHT WEIGHT
DRESP2    100     FWEIGHT 100
DRESP1    10
DEQATN   100     FWEIGHT(ENTWGHT) = ENTWHGT - 998.
```

The second alternative is to assess whether the design variable has an appreciable effect on the design in any case. If the variable does not affect the weight significantly and it has little effect in satisfying a constraint perhaps it should be removed from the design task or perhaps it can be combined with a number of other variables that are negligible by themselves but become significant when they vary together. Combining design variables makes the optimizer's task easier and perhaps makes it easier for you to interpret the results as well. In any case, it is always advisable, prior to optimization, to perform a design sensitivity analysis to ensure that the objective function has sufficient sensitivity with respect to the design variables.

The third, and perhaps, best alternative, is to invoke the OBJMOD parameter on the DOPTPRM entry. This acts to make the objective relative to the starting design.

A related difficulty can occur if you formulate an objective that has zero as its minimum. This can happen when, for example, when you are trying to minimize the error between test results and your finite element



analysis using some type of least squares error function. It could be that the optimizer has no trouble in reducing the error from 1000.0 to 0.01 but then bounces around between .01 and .005. The convergence tests of [Figure 2-17](#) would regard this as a very large change and dictate that further design cycles be made. In fact, this may be an insignificant difference and the extra design cycles are wasted. In this case, it would be helpful to add a constant to the error function at a level that you feel is acceptable. For example, if you think getting to within 0.01 with the error function is a good answer, adding 10.0 to this function will allow the problem to terminate whenever the change in the error is less than .01 (this assumes that the default value of CONV1=0.001 is being used). Alternatively, you could leave the error function unchanged but set the CONV2 parameter to the 0.01 absolute change you regard as indicating a converged solution.



Defining the Constraints

To define constraints in MSC Nastran, you need to

- Identify a first-level response (or group of responses) using a DRESP1 Bulk Data entry, write a second-level response (or group of second-level responses) using a DRESP2 Bulk Data entry, or develop a third-level response (or group of third-level responses) using your own application program.
- If the constraint is to be applied to a DRESP2 quantity that references DRESP1 quantities from different subcases, the DRSPAN case control is required at the subcase level to identify those responses that span subcases.
- Specify the response bounds using a DCONSTR entry.
- Optionally collect the DCONSTR entries into a master set using the DCONADD entry.
- Select DCONSTR and/or DCONADD entry sets in Case Control using either the DESGLB command for “global” (e.g., non-subcase dependent) responses or the DESSUB command for subcase-dependent responses.

In most cases, the constraints are associated with a particular analysis type and subcase, in which case the DESSUB Case Control command is appropriate. In this regard, you should know that MSC Nastran does not allow you to impose subcase dependent constraints that are on responses that are not consistent with the ANALYSIS being performed in that subcase. For example, it is a User Fatal error if you attempt to constrain a LAMA (buckling) response type in an ANALYSIS = STATICS subcase.

The DESGLB request is used in cases where the response is not associated with a subcase. The most obvious example is if you want to constrain a WEIGHT or VOLUME response. If you constrain this response in multiple subcases, you would get duplicate constraints that cause extra work for the optimizer. Instead, it is recommended that you apply this constraint before any subcase using the DESGLB request. Another example where the DESGLB request is appropriate is when you want to specify a relation among the design variables. An example of this given in [Equation \(1-20\)](#) where an aspect ratio limit is imposed on the dimensions of a beam. Since this response is not a function of the finite element analysis, the DESGLB command is used to select this constraint.

The DCONADD Bulk Data entry allows you to collect constraints into an overall set that is selected using the DESSUB or the DESGLB Case Control command. An example where this is useful is a static design task that has multiple subcases. It is likely that the strength limits that are imposed on response types such as STRESS, STRAIN, or CFAILURE are applicable in every subcase. At the same time there may be displacement limits that are unique to each subcase. The availability of the DCONADD entry makes it possible to have a single set of DCONSTR entries for the strength limits. These entries are then combined with the displacement limits using the DCONADD entry.

The DCONSTR entry bounds are used by MSC Nastran to create a pair of normalized constraints, one for the lower bound and one for the upper bound, as discussed in [The Design Model](#). The DCONSTR entry contains an ID, which is invoked either directly by a DESGLB/DESSUB Case Control command or indirectly through a DCONADD entry.

The DCONSTR entry is typically used to provide upper and lower bounds of the constraint, but there are two special features that can be useful in frequency response optimization. The first is that integer IDs



can be used in the fields that specify the bounds. The presence of an integer indicates that the bound is specified by a frequency dependent table. This enables specification of bound as a function of frequency. The second feature is that the DCONSTR entry has optional inputs that specify the lowest and highest frequency for which the constraint is to be applied.



4

Input Data

- Optimizers (Licensing) 133
- File Management 134
- Executive Control 136
- Case Control Section 137
- Bulk Data Entries 143
- Parameters Unique to Design Sensitivity and Optimization 212
- Design Responses 215
- Optimization Parameters 235
- External Response 245
- User Interface for Setting Up External Response 248



This chapter begins with a description of licensing options and then describes all Executive Control, File Management, 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. Overriding of the optimizer's internal parameters
4. Control of the program flow and results output



Optimizers (Licensing)

MSC Nastran provides two optimization options:

1. Design Optimization - Enables both the MSCADS and IPOPT optimizer.
2. MultiOpt Optimization - Enables Multi_Model optimization.

The Design Sensitivity capability described in this guide does not require either option.

MSCADS is MSC's version of the ADS (Automated Design Synthesis) code, a public domain FORTRAN program developed by a predecessor of the Vanderplaats R&D, Inc under contract to NASA/Langley and the US Navy. IPOPT is an open source code from COIN-OR and maintained by IBM.

MSC Nastran selects the optimizer based on logic which estimates which one will provide the better performance. There are two ways to override this default selection. The first is by the OPTCOD parameter (either MSCADS or IPOPT) which can be input on the DOPTPRM entry. The second way is by specifying an executive system parameter as shown in the system cell summary.

Table 4-1 System Cell Summary

System Cell Number	System Cell Name	Description and Default value
413	OPTCOD	<p>Specifies which optimization code is to be used in SOL 200</p> <ul style="list-style-type: none">• 0 Optimizer is selected by the code to achieve the best performance (default)• 3 MSCADS• 4 IPOPT

Note: If one optimizer is selected with SYSTEM(413), and the other optimizer is selected with the OPTCOD parameter on the DOPTPRM entry, the latter is selected.



File Management

Inputs are required in the File Management Section of the input file only in the special circumstances cited in this section.

Shape Optimization

File Management Section statements are required in 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 (see [Relating Design Variables to Shape Changes](#)). The following statements must be used:

The filename file.MASTER is the name of the database for the auxiliary model analysis.

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

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 [Shape Optimization of a Culvert](#) for an example using this method.

User Defined Beam Libraries

MSC Nastran contains a beam library feature that allows you to input beam dimensions for an array of cross section types, such as ROD, I and T. You can augment or replace MSC’s section types with your own sections using an Application Programming Interface (API) option as outlined in [Adding Your Own Beam Cross-Section Library](#). The dimensions of these user defined cross sections can also be designed. The use of these user defined beam sections is enabled by a CONNECT file management statement that identifies the evaluator. The form of the CONNECT statement is:

```
CONNECT BEAMEVAL group evaluator
```

For example:

```
CONNECT BEAMEVAL MYBEAMS MYEVALS
```

where ‘group’ is the group name for the user supplied beam library and is called out on the PBARL or PBEAML entry and must therefore be eight characters or less. ‘Evaluator’ is the evaluator connection file. Each entry in the file uniquely associates the group name with the server program. You can have multiple connect entries for multiple user defined beam libraries and also for the External Responses discussed next.



External Responses

As first discussed in [The Design Model](#), the DRESP3 Bulk Data entry allows you to use an Application Programming Interface (API) to compute external responses that beyond those available using the internal DRESP1 and DRESP2 entries. The use of these user defined external responses is enabled by a CONNECT in much the same fashion as for the user defined beam libraries just described. The form of the CONNECT statement in this cases is:

```
CONNECT DRESP3 group evaluator
```

For example:

```
CONNECT DRESP3 MYRESP MYEXT
```

where ‘group’ is the group name for the external response and is called out on the DRESP3 entry and must therefore be eight characters or less. ‘Evaluator’ is the name of the server program and is currently uniquely associated with the group name. You can have multiple connect entries for multiple external response libraries and also for the user defined beam cross sections discussed above.



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.



Case Control Section

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:

1. Analysis discipline definition
2. Design task definition
3. Design response characterization
4. Shape basis vector computation

This section discusses each of these tasks individually.

Analysis Discipline Definition

Design optimization in MSC 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 in SOL 200:

ANALYSIS = {
STATICS
MODES
BUCKLING
DCEIG
DFREQ
MCEIG
MFREQ
MTRANS
SAERO
FLUTTER}

The following are a few Case Control issues to be aware of when using the ANALYSIS command.

1. If ANALYSIS is specified above the subcase level, all subsequent subcases utilize that analysis type until ANALYSIS is redefined in a later subcase.
2. The buckling subcase defined by ANALYSIS = BUCK must also identify a STATICS subcase that supplies the displacement that is used in the global buckling analysis. The STATIC subcase is selected using the STATSUB Case Control command (STATSUB is only used with ANALYSIS = BUCK and is not available for other analysis types).
3. Multiple boundary conditions may be used in Solution 200 for ANALYSIS = STATICS, MODES, BUCKLING, DFREQ, MFREQ, SAERO and FLUTTER.



4. Only a single subcase is supported for ANALYSIS = DCEIG,MCEIG and MTRAN.
5. In general, any number of analysis types can be selected in a single run, but ANALYSIS=DFREQ and MFREQ cannot both be used in the same run.
6. [Table 2-5](#) indicates which disciplines support multiple subcases and boundary conditions.

Design Task 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:

DESOBJ($\frac{\text{min}}{\text{max}}$) = n

where n is the set identification of a design response on either a DRESP1, DRESP2, or DRESP3 Bulk Data entry. This response must be a single, scalar quantity.

A DESOBJ command 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.

A DESOBJ command is required in most cases, but there are certain situations where it is not needed:

1. If only sensitivity results are of interest, a DESOBJ command is not required.
2. If Fully Stressed Design is being used, there is not a requirement for a DESOBJ command. However, it is recommended that the DESOBJ command be used and that it point to a WEIGHT response type so that the weight response will be calculated and reported.

Note: If a DESOBJ request is needed for optimization and is not provided, the error is not detected until optimization is attempted and therefore after an analysis and sensitivity analysis has been performed. This can be an expensive error if these analyses consume significant resources.

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.

Subcase Spanning Responses

A special situation occurs when you wish to combine responses from multiple subcases. For example, you may want to constrain the average displacement across multiple subcases. The constraint is on a global quantity in this case, so that the **DESGLB** command just discussed is used above the subcase level. It is also necessary to identify the subcase responses that are to be included in the subcase spanning response. This is done with a DRSPAN command:

```
DRSPAN = n
```

The n is the set identification of a previously appearing SET case control command.

The DRSPAN request occurs at the subcase level and must appear in every subcase that contains DRESP1 quantities that are to be used in the DRESP2 that spans subcases.

Design Variable Set Selection

The **optional** DESVAR Case Control command selects a set of design variables defined by a SET command. If the command is absent, all DESVAR Bulk Data entries will be utilized in the design task. A maximum of one DESVAR Case Control command can appear and it must appear above the subcase level.

The basic purpose of the DESVAR command is to address the scenario where the design task started with many design variables, but it is desired to focus on a subset. This could be done by editing the bulk data file to remove the unwanted DESVAR entries, but this is a tedious process that is avoided using the DESVAR case control command. It is still necessary to impose rules on what happens when these unneeded DESVAR's are referenced on other bulk data entries. These rules include:

- If a DVxRELY (e.g., DVPREL2) or DLINK entry references DESVARs that are all included in the set specified in case control, then that property is designed.
- If a DVxRELY of DLINK entry references DESVARS that not included in the set specified in case control, then that property is not designed
- If a DVxRELY or DLINK entry references some DESVARs that are all included in the set specified in case control and some that are not, a User Fatal Message is produced and the run is terminated.
- If a DRESP2 or DRESP3 references any DESVARs that are not included in the set specified in case control, a User Fatal Message is produced and the run is terminated.



The DESVAR command is:

Format:

DESVAR = n

The n is the set identification of a previously appearing SET Case Control command.

Design Sensitivity Print

The meaning and importance of design sensitivity has been discussed at length in [Design Sensitivity Analysis](#). The design sensitivity data is quite useful in its own right and can be printed using the DSAPRT Case Control command.

$$\text{DSAPRT} \left[\begin{bmatrix} \text{FORMATTED} \\ \text{UNFORMATTED} \\ \text{NOPRINT} \end{bmatrix}, \begin{bmatrix} \text{NOEXPORT} \\ \text{EXPORT} \end{bmatrix}, [\text{START} = i], [\text{BY} = y], [\text{END} = k] \right] = \begin{bmatrix} \text{ALL} \\ n \\ \text{NONE} \end{bmatrix}$$

This command supports a number of options, the first of which indicates whether the output is to be formatted. The formatting provides labeling information and is usually the preferred option. The data can be exported to an external file using the EXPORT option. This enables the use of the sensitivity data in another application. The frequency of the output is controlled by three parameters: START, BY, and END. These parameters refer to the design cycle so that START indicates after which design cycle the print is to begin, BY indicates how often it is to be printed between the START and END cycles and END identifies the final cycle. The right-hand side of the command identifies the set of response IDs that are to be printed. The default of ALL prints all the sensitivities.

Note: A response sensitivity is available for printing only if the response has been constrained and then only if the constraint has survived the screening process described in [Constraint Screening](#). The DSCREEN Bulk Data entry described in [Bulk Data Entries](#) provides a way of forcing the retention of responses that are of interest.

Mode Tracking

A normal modes subcase (ANALYSIS = MODES) can constrain the normal modes eigenvalues or frequencies based on mode number. As the design changes, the order of the modes may change so that the physical nature of the mode being constrained would change as well. As discussed in the Mode Tracking paragraphs of [Multidisciplinary Analysis](#), MSC Nastran can be made to “track” the modes so that the constraint is being applied to the same physical mode as the design changes. This is done using a MODTRAK Case Control command:

MODETRAK = n

where n invokes the ID of a MODTRAK Bulk Data entry.



Residual Vectors

The RESVEC Case Control command is used to augment the modes created from a normal modes analysis with additional shapes tailored to the modal analysis task at hand. The addition of these modes can greatly improve the accuracy of the finite element analysis and occurs automatically if this command is not specified. The RESVEC Case Control command can be used to control the creation of these additional modes and one of the describers of this command is relevant here.

ADJLOD/NOADJLOD is an option on the RESVEC command that indicates whether adjoint loads should be used to create residual vectors. These loads are derived from DRESP1 entries that invoke grid responses in a frequency or transient response analysis. The ADJLOD option (the default) creates these residual vectors and is recommended, while use of the NOADJLOD option is available to block the inclusion of adjoint loads in residual vector computations.

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.

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 MSC 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 output commands are sometimes used to resolve design response ambiguities. For example, element plate stresses can be output using either von Mises or maximum shear. (The ambiguity arises since both share the same item code ID referenced on the DRESP1 entry.) The form identified in Case Control is also used 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).

Note: The SET number (15 in the example above) has no effect on which responses are used in the design.

A similar convention applies to the location and type of stress responses that are used as design responses for plate elements. The element responses have options [CENTER, CUBIC, SGAGE, (CORNER or BILIN)].

A DRESP1 which references a plate element will have responses that are driven by what has been specified in case control. This is particularly important when selecting corner versus element center responses since the item codes are dependent on which option is selected.



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,

STRESS(IMAG)	= ALL	(real/imaginary form for analysis and optimization)
STRESS(PHASE)	= ALL	(magnitude/phase form for analysis and optimization)

Shape Basis Vector Computation

Additions to Case Control for shape optimization are only necessary when using the Analytic Boundary Shapes method.

[Relating Design Variables to Shape Changes](#) shows 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 in [Relating Design Variables to Shape Changes](#).

Generation of a New Bulk Data File

The ECHO Case Control command has an option that is applicable only in design optimization:

ECHO = PUNCH (NEWBULK)

It is available to provide an unsorted Bulk Data file following a design optimization task that has updated entries replacing the original entries. For example, the input XINIT value on a DESVAR entry is replaced by the final value for this design variable from the design task. Similarly, new thicknesses that are produced will be placed on the updated PSHELL entry. This file is contained in the output .pch file that is produced by MSC Nastran. A complete description of the ECHO Case Control command is available in [ECHO](#) in the *MSC Nastran Quick Reference Guide*. Additional information on the output produced by the NEWBULK option can be found in [Special Punch Considerations for Topology Optimization](#). An innovative use of the NEWBULK parameters is to use it in conjunction with PARAM OPEXIT 1 through 3 to get a new bulk data deck that reflects the changes introduced by the design model.



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. This section contains a brief listing and description of all the Bulk Data entries related to design sensitivity and optimization. This information is the alphabetical order of the entry name and has been extracted from the complete listings in the [The Bulk Data Section](#) (p. 1097) in the *MSC Nastran Quick Reference Guide*. For each entry in this section, related entries have also been listed with appropriately shaded fields to help clarify the interrelations among the various entries.

BEADVAR

Topography Design Variable

Purpose:

Defines design region for topography (bead or stamp) optimization.

Entry Description:

	1	2	3	4	5	6	7	8	9	10
BEADVAR	ID	PTYPE	PID	MW	MH	ANG	BF	SKIP		
	"DESVAR	"NORM/XD	YD	ZD	CID	XLB	XUB	DELXV		
	"GRID	"NGSET	DGSET							

Example using NORM:

BEADVAR	IO	PSHELL	101	10.0	20.0	70				
---------	----	--------	-----	------	------	----	--	--	--	--

Example using "DESVAR" and "GRID":

BEADVAR	10	PPSHLL	101	10.0	20.0	70.0		NONE
	DESVAR	2.0	3.0	4.0		-1.0	1.0	
	GRID	100						

Field	Contents
ID	Unique topography design region identification number. (Integer > 0)
PTYPE	Property entry type. Used with PID to identify the element nodes to be designed. (Character: "PSHELL", "PSHEAR", "PCOMP", or "PCOMPG".)
PID	Property entry identifier. See Remark 1. (Integer > 0)
MW	Minimum bead width. This parameter controls the width of the beads. The recommended value is between 1.5 and 2.5 times the average element width. See Remark 2. (Real > 0.0)



Field	Contents
MH	Maximum bead height (Real > 0.0). This parameter sets the maximum height of the beads when XUB=1.0 (as Default). See Remark 2.
ANG	Draw angle in degrees (0.0 < Real < 90.0). This parameter controls the angle of the sides of the beads. The recommended value is between 60 and 75 degrees.
BF	Buffer zone ('yes' or 'no'; Default='yes'). This parameter creates a buffer zone between elements in the topography design region and elements outside the design region when BF='yes'. See Remark 3.
SKIP	Boundary skip ("bc", "load", "both", or "none"; Default = "both"). This parameter indicates which element nodes are excluded from the design region. "bc" indicates all nodes referenced by "SPC" and "SPC1" are omitted from the design region. "load" indicates all nodes referenced by "FORCE", "FORCE 1", "FORCE2", "MOMENT", "MOMENT 1", "MOMENT2", and "SPCD" are omitted from the design region. "both" indicates nodes with either "bc" or "load" are omitted from the design region. "none" indicates all nodes associated with elements referencing PID specified in field 4 are in the design region.
"DES VAR"	Indicates that this line defines bead design variables that are automatically generated.
NORM/XD, YD, ZD	Bead vector (draw direction). Norm indicates the shape variables are created in the normal directions to the elements. If XD, YD, and ZD are provided, the shape variables are created in the direction specified by the xyz vector defied by XD/YD/ZD that is given in the basic coordinate system or CID. See Remark 4. (Character or Real, Default = blank = norm).
CID	Coordinate system ID used for specifying draw direction (Blank or Integer > 0; Default = blank = basic coordinate system)
XLB	Lower bound. (Real < XUB or blank; Default = blank=0.0). This ensures the lower bound on grid movement equal to XLB*MH. See Remark 5.
XUB	Upper bound. (Real > XLB or blank; Default = 1.0). This sets the upper bound of the beads equal to XUB*MH. See Remark 5.
DELXV	Fractional change allowed for the design variable during approximate optimization. See Remark 3. (Real > 0.0; Default = 0.2)
"GRID"	"Indicates this line defines what element nodes can be added and/or removed from topography design regions.
NGSET	All grids listed on a Bulk Data entry SET1 = NGSET are removed from topography design regions.
DGSET	All grids listed on a Bulk Data entry SET1 = DGSET are added to topography design regions.

Discussion:

1. Multiple BEADVAR's are allowed in a single file. Combined topometry, topology, topography, sizing, and shape optimization is supported in a single file.



The user can provide allowable bead dimensions.

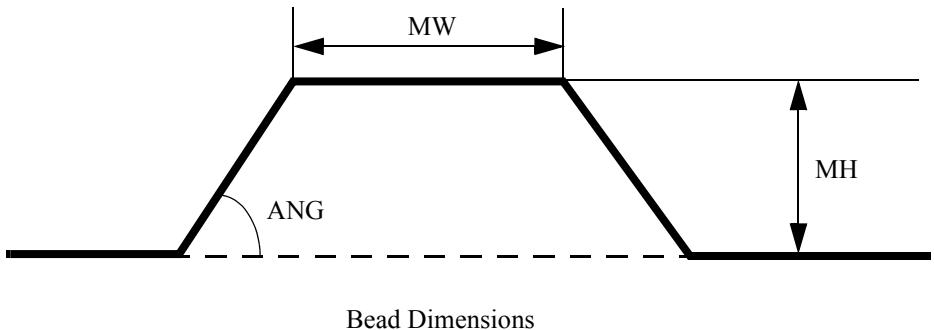
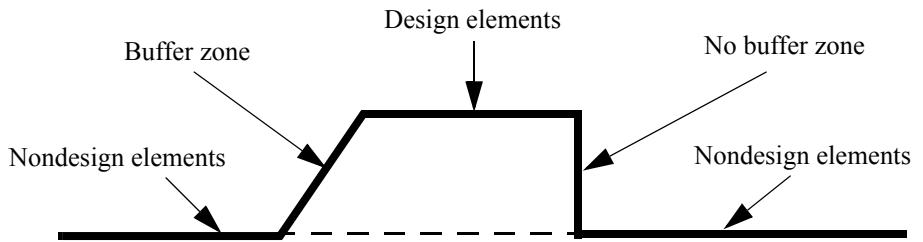
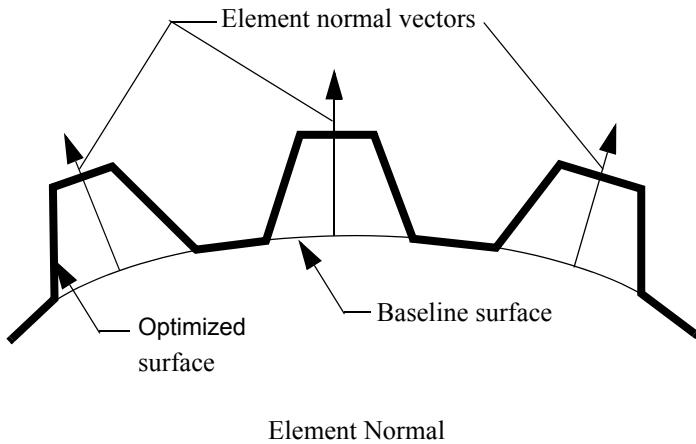


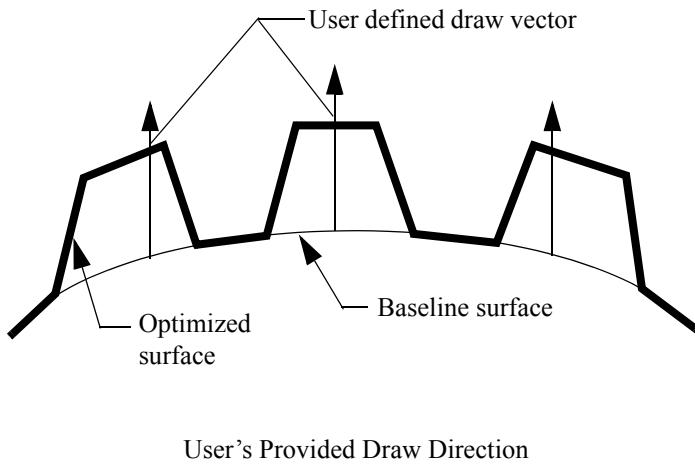
Figure 4-1 Bead Dimensions

It is recommended to set buffer zone = yes to maintain a good quality of mesh during topography optimization.

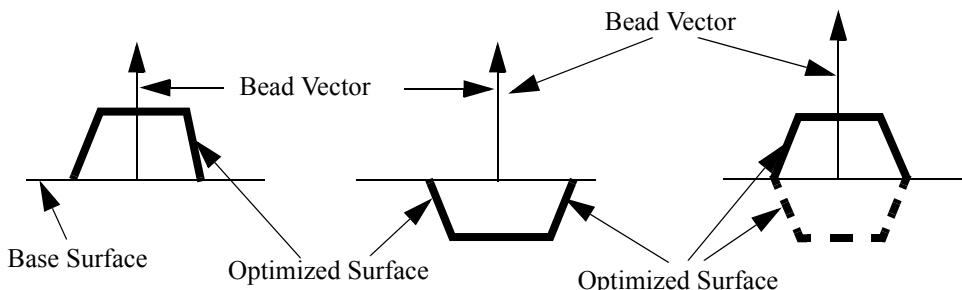


The grids move in the normal direction. All element grids referenced by one BEADVAR entry must follow the right hand rule.





To force the grids to move only in the positive bead vector direction (one side of the surface), use $XLB = 0.0$. To force the grids to move only in the negative bead vector direction (another side of the surface), use $XUB = 0.0$. To allow grids to move in both positive and negative bead vector directions, use $XLB < 0.0$ and $XUB > 0.0$. For example,



The jobname.op2 (with setting PARAM, POST, -1) has topography results (shape change) that can be viewed by Patran. The text file jobname.pch also has updated grid coordinates that can be copied to the original file, replace the original grids, and imported to Patran and other post-processors to view topography optimization results.

BNDGRID

Purpose:

Specifies a list of grid point identification numbers on design boundaries or surfaces for shape optimization.



Entry Description:

BNDGRID	C	GP1	GP2	GP3	GP4	GP5	GP6	GP7	
	GP8	-etc.-							

Field	Contents
C	Component number (any unique combination of integers 1 through 6 with no embedded blanks).
GPi	Shape boundary grid point identification number.

Discussion:

[Relating Design Variables to Shape Changes](#) identifies four alternative methods for defining shape basis vectors and two of these methods: Geometric Boundary Shapes and Analytic Boundary Shapes entail first defining the shape basis vector on the boundary. This, in turn, requires the definition of which degrees-of-freedom appear on the boundary. The BNDGRID entry performs this task by selecting a component set and then identifying the associated grid points. [Analytic Boundary Shape](#) provides an example that applies the Analytic Boundary Shapes Method and makes use of the BNDGRID entry.

Note: Grid coordinates that appear on the BNDGRID entry but that are not affected by the displacement from the auxiliary model are assumed to not change their location during the shape optimization task.

DCONADD**Purpose:**

Defines design constraints 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	LOWFQ	HIGHFQ			
---------	------	-----	--------	--------	-------	--------	--	--	--

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/ LID	UALLOW/ UID	LOWFQ	HIGHFQ			
---------	------	-----	----------------	----------------	-------	--------	--	--	--

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)
LID	Set identification of a TABLEDi entry that supplies the lower bound as a function of frequency.
UALLOW	Upper bound on the response quantity. (Real, Default = 1.0E20)
UID	Set identification of a TABLEDi entry that supplies the upper bound as a function of frequency.
LOWFQ	Low end of frequency range in Hertz. (Real \geq 0.0, Default = 0.0)
HIGHFQ	High end of frequency range in Hertz. (Real \geq LOWFQ, Default = 1.0E+20)

Associated Entries:

The IDs on DRESP1, DRESP2 and DRESP3 are the RIDs called out on the DCONSTR entry.



DRESP1	ID	LABEL	RTYPE	PTYPE	REGION	ATTA	ATTB	ATT1	
	ATT2	-etc.-							

DRESP2	ID	LABEL	EQID	REGION					
	"DESVAR"	DVID1	DVID2	-etc.-					
	-	-	-	-					
	"DRESP1"	NR1	NR2	-etc.-					
	"DRESP2"	NRR1	NRR2						
	-	-	-	-					

DRESP3	ID	LABEL	GROUP	TYPE	REGION				
	"DESVAR"	DVID1	DVID2	-etc.-					
	-	-	-	-					
	"DRESP1"	NR1	NR2	-etc.-					
	"DRESP2"	NRR1	NRR2						
	-	-	-	-					

The DCONADD entry can be used to form the union of a number of DCONSTR sets.

DCONADD	DCID	DC1	DC2	DC3	-etc.-				
---------	------	-----	-----	-----	--------	--	--	--	--

Discussion:

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 a DRESP1, DRESP2 or DRESP3 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](#). The LOWFQ and HIGHFQ fields are only functional for response types with the 'FR' prefix, for example FRDISP and excepting RTYPE=FREQ. In these situations the bounds are only applied to the response if the excitation frequency falls with the range of these two frequencies in Hz. LOWFQ and HIGHFQ are ignored for other responses types. The upper and lower bounds can be specified in one of two ways. If real numbers are used in the bound field, these are the bounds. If integers are used, these point to TABLEDi entries that enable the input of these bounds as a function of frequency.

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.

There are two guidelines for the user of the DCONSTR entry that can be noted here. The first is that it is recommended that zero (and near zero) values be avoided when specifying LALLOW and UALLOW. There are two reasons for this. The first is that this causes problem in the normalization as explained in



[Design Constraints](#). If the constraint limit really is zero (for example it is desired to restrict the flutter damping to be less than zero), it is recommended that the response be recast using a DRESP2 entry to offset the response by some amount. For example, if a value of 1.0 is added to the flutter damping response, it will now be possible to place an upper bound of 1.0 on the DRESP2 response. The second difficulty with a zero response is that it can lead to unintended consequences in terms of constraint screening. Suppose the response is the von Mises stress and it is desired that these stresses never exceed 30KSI. The UALLOW value is then 30000.0 but it may seem natural to set LALLOW to 0.0. The unintended consequence of this is that this may create a critical constraint when the element is essentially unstressed. The optimizer will try and satisfy this constraint with unpredictable results. In this case, it is recommended that the LALLOW field be left blank so that the default value of -1.0E20 will be used and this will never be a critical constraint.

The second guideline is that is recommended that “equality” constraints allow some tolerance as also discussed in [Design Constraints](#). For example, if you are trying to redesign the structure so that the first normal mode is exactly 6.0 Hz., you should specify limits of LALLOW = 5.9 Hz. and UALLOW = 6.1 Hz. so that the optimizer has a chance to satisfy both limits simultaneously. If the limits were both set to 6.0, one or the other constraint would always be violated.



DDVAL

Purpose:

In a discrete optimization task, provide a list of discrete values that a design variable can assume.

Entry Description:

DDVAL	ID	DDVAL1	DDVAL2	DDVAL3	DDVAL4	DDVAL5	DDVAL6	DDVAL7	
	DDVAL8	-etc.-							

Field	Contents
ID	Unique discrete value set identification number. (Integer > 0)
DDVALi	Discrete values. (Real)

Associated Entries:

The ID on the DDVAL entry is called out the DDVAL field on the DESVAR entry:

DESVAR	ID	LABEL	XINIT	XLB	XUB	DELXV	DDVAL		
--------	----	-------	-------	-----	-----	-------	-------	--	--

Discussion:

The Discrete Variable Optimization capability discussed in [Special Topics](#) enables the designation of design variable values from a discrete set of values. This selection is based on the conventional continuous optimization that is the main topic of this User's Guide and provides a means of performing optimization when the application requires the design variable take on a discrete value (perhaps from available sheet metal thickness or, in a composite design, an integer number of plies).

The DDVAL entry provides the allowable set of discrete values and is invoked by one or more design variables. The real numbers can be input in arbitrary order and the THRU and BY features provide convenient way of specifying a series of numbers.

If a DESVAR entry does not have a value in the DDVAL field, the design variable is allowed to vary continuously in a discrete variable optimization task.

If the range of discrete values for a DESVAR entry do not extend out to the XLB and XUB range specified on the DESVAR entry, the XLB and XUB values are adjusted to extend only over the discrete values. Similarly, a discrete value that is outside the user specified XLB/XUB range will never be selected by the discrete optimization algorithm.



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.-					
	-	-	-	-					
	"DRESP1"	NR1	NR2	-etc.-					
	-	-	-	-					
	DRESP2	NRR1	NRR2						

DVPREL2	ID	TYPE	PID	FNAME	PMIN	PMAX	EQID		
	"DESVAR"	DVID1	DVID2	DVID3	-etc.-				
	"DTABLE"	LABL1	LABL2	LABL3	-etc.-				

Discussion:

A powerful feature of MSC Nastran design sensitivity and optimization is that it allows you 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, such as SIN or MAX, may be used.

The equation input arguments are formal arguments that are defined at runtime. The DRESP2 entry for responses and the DVCREL2, DVMREL2, DVPREL2 entries for design variable-to-property relations specify the input, or actual, arguments to the appropriate DEQATN entry. A nested form of equation definition may 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 power and versatility of the DEQATN, particularly when used with the DRESP2 entry, has given rise to a number of special requests for this entry to extend the ease-of-use of the DEQATN. The features that have been provided in response to these requests are explicitly pointed out here because they have primary applicability in design sensitivity and optimization.

Special Functions

The majority of the intrinsic functions used in the DEQATN are those that are available in a programming language such as FORTRAN; e.g., ABS for absolute value and TAN for tangent. However, there are eight functions as shown in [Table 4-2](#) that have been provided, primarily, to give support in dynamic response design and are applicable for any relevant analysis. The SUM,RSS, SSQ and AVG functions provide a shorthand operation for operating on a series of arguments. (Refer to the DRESP2 entry for an alternative means of performing these operations and to the ATTB field of the DRESP1 entry for the frequency response RTYPES for another simplified way of inputting these operations in special cases.) The use of these operators on the DEQATN is not limited to responses so that you could use the AVG function to find, for example, the average of a set of design variables). The DB,DBA, INV and INVDBA provide a simplified way of dealing with acoustic responses.

Table 4-2 Special DEQATN Functions

Format	Description	Mathematical Expression
AVG((X_1, X_2, \dots, X_n))	average	$\frac{1}{n} \sum_{i=1}^n X_i$
DB(P, PREF)	sound pressure in decibel	$20.0 \cdot \log\left(\frac{P}{PREF}\right)$
DBA(P, PREF, F)	sound pressure in decibel (perceived)	$20.0 \cdot \log\left(\frac{P}{PREF}\right) + 10.0 \cdot \log(Ta1) + 10.0 \cdot \log(Ta2)$
INVDB(DB, PREF)	inverse Db	$10^{\left(\frac{DB + \log PREF}{20.0}\right)}$
INVDBA(DBA, PREF, F)	inverse Dba	$10^{\left(\frac{(DBA - 10.0 \cdot \log(Ta1) - 10.0 \cdot \log(Ta2))}{20.0}\right)}$
RSS (X_1, X_2, \dots, S_n)	square root of sum of squares	$\sqrt{\sum_{i=1}^n X_i^2}$



Table 4-2 Special DEQATN Functions (continued)

Format	Description	Mathematical Expression
SSQ (X_1, X_2, \dots, S_n)	sum of squares	$\sum_{i=1}^n X_i^2$
SUM (X_1, X_2, \dots, S_n)	summation	$\sum_{i=1}^n X_i$

X_1, X_2, \dots, S_n, P = structure responses or acoustic pressure

PREF = reference pressure

F = forcing frequency

DB = acoustic pressure in Decibel

DBA = perceived acoustic pressure in Decibel

$$Ta1 = \frac{K3 \cdot F^4}{(F^2 + P2^2)(F^2 + P3^2)}$$

$$Ta2 = \frac{K1 \cdot F^4}{(F^2 + P1^2)^2 (F^2 + P4^2)^2}$$

$$K1 = 2.242882e+16$$

$$K3 = 1.562339$$

$$P1 = 20.598997$$

$$P2 = 107.65265$$

$$P3 = 737.86223$$

$$P4 = 12194.22$$

Implied Forcing Frequency Argument

Frequency input to synthetic relations can be supplied in an implicit format that is activated by:

1. Frequency response-type ("FR" or "PSD") arguments for the DRESP1 IDs invoked by a DRESP2.
2. A DEQATN entry whose input argument list exceeds the number of arguments defined on the referencing DRESP2 entry.

The situation is perhaps most easily described by way of an example. Assume a simple average is to be computed based on frequency response-computed displacements that are "normalized" by their respective forcing frequencies. We can then write:



```
$RESP1, ID, LABEL, RTYPE, PTYPE, REGION, ATTA, ATTB, ATT1, +
$, ATT2, ...
dresp1, 10, fdisp1, frdisp, , , 1, 10., 1001
dresp1, 20, fdisp2, frdisp, , , 1, 20., 1001
$RESP2, ID, LABEL, EQID, REGION, , , , ,
$, DRESP1, NR1, NR2, , , , , ,
dresp2, 30, avgfd, 100, , , , ,
+, dresp1, 10, 20
+,
deqatn 100 avg(d1,d2,f1,f2) = (d1/f1 + d2/f2)*0.5
```

Normally, the number of input arguments and values specified on the DEQATN/DRESP2 entries must match. The rules, however, are relaxed when frequency response quantities (i.e., those preceded by "FR") are encountered as input to the equation. In such cases, additional arguments can be defined via implicit reference to the list of frequency values appearing on the DRESP1 entries, in the encountered order. In the above example, f1 and f2 are 10. hertz and 20. hertz, respectively.

The above capability is particularly useful in the case of frequency definitions via the FREQ3, FREQ4, and FREQ5 entries. Again, a simple example explains this feature. The presence of FREQ3, FREQ4, or FREQ5 entries is assumed in the following example:

```
DRESP1,10,FRGD,FRDISP,,,1,,21
DRESP2,101,DBA,201,
,DTABLE,PREF,,,
,DRESP1,10
DTABLE,PREF,1.0
DEQATN 201 X(PREF,P,F)=DBA(P,PREF,F)
```

A DRESP1/DRESP2 pair of responses will be generated for every forcing frequency, the value of which is generally unknown at the outset. Note that the third input to the equation, F, is in excess of the number of arguments defined in the DRESP2 relation. Because a frequency response displacement is referenced, the code will automatically assign its value to be the corresponding forcing frequency.

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.

The optional DESVAR Case Control command can be used to specify the set of DESVAR Bulk Data entries that are to be used in the design task. If the DESVAR Case Control command is absent, all DESVAR Bulk Data entries will be used.



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, $XLB \leq XINIT \leq XUB$)
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, DVCREL1, DVMREL1, DVLREL1 or DVPREL2, DVCREL2, DVMREL2 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 and external responses using the DRESP3 entry. The DVCRELi and DVMCRELi entries are not shown here because of their close similarity with the DVPRELi entries.

DVPREL1	ID	TYPE	PID		PMIN	PMAX	C0		
	DVID1	COEF1	DVID2	COEF2	DVID3	-etc.-			

DVPREL2	ID	TYPE	PID		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.-					
-	-	-	-	-					
	“DRESP1”	NR1	NR2	-etc.-					
-	-	-	-	-	-	-			

DRESP3	ID	LABEL	GROPU	TYPE	REGION				
	“DESVAR”	DVID1	DVID2	-etc.-					
-	-	-	-	-					
	“DRESP1”	NR1	NR2	-etc.-					
-	-	-	-	-					
USRDATA	GEOMETRY DATA FILE								

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. Design variables can also be used as design parameters that are unrelated to the structural properties. In this case, they are invoked by DRESP2 or DRESP3 entries. The DESVAR entry is used to define an individual design variable.

The DELXV attribute is used to specify the amount a design variable can change during one optimization cycle. This relates to the [The Approximate Model](#) and should be used when it appears that the results from the approximate analysis are not closely matched by the subsequent exact analysis. DELXV was provided primarily for shape optimization tasks but is applicable in any context.

The DDVAL field enables the Discrete Variable Optimization capability of [Special Topics](#).

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)
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:

$$x_D = c_o + \sum_i c_i x_i$$

x_D = dependent design variable

c_o = constant

c_i = constant multiplying coefficient

x_i = independent design variables

Design variable linking can be used, for example, to ensure structural symmetry or unify sizing changes across property groups. In this way, you can perform “what-if” studies to see how different modeling assumptions affect the derived design. 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 The Bulk Data Section in the <i>MSC Nastran Quick Reference Guide</i> . (Character)
VALi	Value of the parameter. (Real or Integer; see the DOPTPRM listing in The Bulk Data Section in the <i>MSC Nastran Quick Reference Guide</i> .)

Associated Entries:

None

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 may be changed using the DOPTPRM entry. An overview of *selected* parameters, grouped according to their functionality is presented in [Optimization Parameters](#).

DRESP1

Purpose:

Defines direct, or first-level, analysis responses to be used in design sensitivity and optimization. The responses identified here are those which are directly available from the analysis results as opposed to second-level responses which are defined using DRESP2 and DEQATN entries or third-level responses which are defined using DRESP3 and an external response evaluator.

Entry Description:

DRESP1	ID	LABEL	RTYPE	PTYPE	REGION	ATT1	ATTB	ATT2	
	ATT2	-etc.-							



Field	Contents
ID	Unique entry identifier. (Integer > 0)
LABEL	User-defined label. (Character)
RTYPE	Response type. See the DRESP1 listing in The Bulk Data Section (p. 1097) in the <i>MSC Nastran Quick Reference Guide</i> . (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.

Associated Entries:

A response identified on a DRESP1 entry may be used either as a constraint or an objective. For constraints, the response is identified by its ID number on DCONSTR entries.

DCONSTR	DCID	RID	LALLOW	UALLOW				
---------	------	-----	--------	--------	--	--	--	--

For the objective, a first-level response must define 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

$$\text{DESOBJ} \begin{pmatrix} \min \\ \max \end{pmatrix} = n$$

where n is the DRESP1 entry ID. (Note that the n could optionally point to a DRESP2 or DRESP3 ID.)

A DRESP1 response can also be used as input to synthetic responses defined on DRESP2 entries or by an external response on a DRESP3. It is referenced by its response ID.

DRESP2	ID	LABEL	EQID	REGION				
-	-	-	-	-				
"DTABLE"	LABL1	LABL2	-etc.-					
"DRESP1"	NR1	NR2	-etc.-					
-	-	-	-	-	-			



DRESP3	ID	LABEL	GROUP	TYPE	REGION				
	"DESVAR"	DVID1	DVID2	-etc.-					
	"DRESP1"	NR1	NR2	-etc.-					

If a DRESP1 is to be used on a DRESP2 or DRESP3 that combines responses from two or more subcases, the DRSPAN Case Control command is required at the subcase level to indicate this:

DRSPAN = n

where n is a previously appearing Set command that lists the DRESP1 IDs.

Discussion:

Depending on the particular analysis discipline, responses such as displacements, stresses, eigenvalues, etc., will be computed. 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 or passed to a third-level, or external, response on a DRESP3 entry for application in a user defined response.

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. An exception are special cases previously discussed in Design Response Characterization.

Every DRESP1 entry has a unique ID, a user defined label and a response type. The response type identifies what type of response quantities is of interest. [Table 4-3](#) identifies each of the 40 unique response types, indicates which ANALYSIS discipline (as defined on the ANALYSIS case control command) they are applicable for, the approximation method (direct or inverse) that is used when APRCOD=2 and the default REGION. If a response type is inconsistent with the ANALYSIS command as given in the table for a particular subcase, it is a User Fatal Error and the run is terminated.

The REGION attribute on this entry is used in the constraint screening process discussed in [Constraint Screening](#). If this field is left blank, the defaults of [Table 4-3](#) apply. The N/A designations in this table refer to scalar responses where the concept of default REGION is not applicable. If a REGION is set, all responses that share the REGION ID and have the same response type will be grouped together during the screening process.

The fact that this single entry supports so many different response types makes this entry rather complex. A complete discussion explaining each of the response types in turn is given in [Design Responses](#).

Table 4-3 Response Type Options on the DRESP1 Entry

Response Types	Associated ANALYSIS Type	Approximation Technique	Default Region
WEIGHT	Any or None	Direct	N/A
VOLUME	Any or None	Direct	N/A
EIGN	MODES	Rayleigh Quotient*	N/A
CEIG	DCEIG or MCEIG	Inverse	N/A
FREQ	MODES	Rayleigh Quotient*	N/A
LAMA	BUCK	Direct*	N/A
DISP	STATICS or SAERO	Inverse	DRESP1 ID
STRAIN	STATICS or SAERO	Inverse	Property ID or N/A
ESE	STATICS or SAERO	Inverse	Property ID or N/A
STRESS	STATICS or SAERO	Inverse	Property ID or N/A
FORCE	STATICS or SAERO	Direct	Property ID or N/A
SPCFORCE	STATICS or SAERO	Inverse	DRESP1 ID
CSTRAIN	STATICS or SAERO	Inverse	Property ID or N/A
CSTRESS	STATICS or SAERO	Inverse	Property ID or N/A
CFAILURE	STATICS or SAERO	Inverse	Property ID or N/A
CSTRAT	STATICS or SAERO	Inverse	Property ID or N/A
TOTSE	STATICS or SAERO	Inverse*	N/A
GPFORCE	STATICS or SAERO	Inverse	DRESP1 ID
GPFORCP	STATICS or SAERO	Inverse	DRESP1 ID
FRDISP	DFREQ or MFREQ	Inverse	DRESP1 ID
PRES	DFREQ or MFREQ	Inverse	DRESP1 ID
FRVELO	DFREQ or MFREQ	Inverse	DRESP1 ID
FRACCL	DFREQ or MFREQ	Inverse	DRESP1 ID
FRSPCF	DFREQ or MFREQ	Inverse	DRESP1 ID
FRSTRE	DFREQ or MFREQ	Inverse	Property ID or Element ID
FRFORC	DFREQ or MFREQ	Inverse	Property ID or Element ID
PSDDISP	DFREQ or MFREQ	Inverse	DRESP1 ID
PSDVELO	DFREQ or MFREQ	Inverse	DRESP1 ID
PSDACCL	DFREQ or MFREQ	Inverse	DRESP1 ID
RMSDISP	DFREQ or MFREQ	Inverse	N/A
RMSVELO	DFREQ or MFREQ	Inverse	N/A



Table 4-3 Response Type Options on the DRESP1 Entry (continued)

Response Types	Associated ANALYSIS Type	Approximation Technique	Default Region
RMSACCL	DFREQ or MFREQ	Inverse	N/A
TDISP	MTRAN	Inverse	DRESP1 ID
TVELO	MTRAN	Inverse	DRESP1 ID
TACCL	MTRAN	Inverse	DRESP1 ID
TSPCF	MTRAN	Inverse	DRESP1 ID
TSTRE	MTRAN	Inverse	Property ID or Element ID
TFORC	MTRAN	Inverse	Property ID or Element ID
TRIM	SAERO	Inverse	N/A
STABDER	SAERO	Inverse	N/A
DIVERG	SAERO	Inverse	N/A
FLUTTER	FLUTTER	Inverse	DRESP1 ID
COMP	STATICS	Direct*	N/A
FRMASS**	ANY	Direct	N/A
ACPWR	DFREQ or MFREQ	Inverse	DRESP1 ID
ACINTS	DFREQ or MFREQ	Inverse	DRESP1 ID
AFPRES	DFREQ or MFREQ	Inverse	DRESP1 ID
AFINTS	DFREQ or MFREQ	Inverse	DRESP1 ID
AFVELO	DFREQ or MFREQ	Inverse	DRESP1 ID
AFPWR	DFREQ or MFREQ	Inverse	DRESP1 ID
STMONP1	STATICS or SAERO	Inverse	DRESP1 ID
STMOND1	STATICS or SAERO	Inverse	DRESP1 ID
MONPNT3	STATICS or SAERO	Inverse	DRESP1 ID
AEMONP1	SAERO	Inverse	DRESP1 ID
AEMOND1	SAERO	Inverse	DRESP1 ID
ERP	DFREQ or MFREQ	Direct	DRESP1 ID
FATIGUE	STATICS	Inverse	DRESP1 ID
DIVERG	SAERO	Inverse	N/A
ABSSTRESS	STATICS and SAERO	Inverse	Property ID or N/A

Table 4-3 Response Type Options on the DRESP1 Entry (continued)

Response Types	Associated ANALYSIS Type	Approximation Technique	Default Region
WMPID	ANY	Direct	N/A

*Note that the approximation technique for these responses can be changed using the ATTB field on the DRESP1 entry.

**FRMASS is only available for topology optimization.



DRESP2

Purpose:

Defines equation responses that are used in the design, either as constraints or as an objective.

Format:

1	2	3	4	5	6	7	8	9	10
DRESP2	ID	LABEL	EQID or FUNC	REGION	METHOD	C1	C2	C3	
"DESVAR"	DVID1	DVID2	DVID3	DVID4	DVID5	DVID6	DVID7		
	DVID8	-etc.-							
"DTABLE"	LABL1	LABL2	LABL3	LABL4	LABL5	LABL6	LABL7		
	LABL8	-etc.-							
"DRESP1"	NR1	NR2	NR3	NR4	NR5	NR6	NR7		
	NR8	-etc.-							
"DNODE"	G1	C1	G2	C2	G3	C3			
	G4	C4	etc.						
"DVPREL1"	DPIP1	DPIP2	DPIP3	DPIP4	DPIP5	DPIP6	DPIP7		
	DPIP8	DPIP9	-etc.-						
"DVCREL1"	DCIC1	DCIC2	DCIC3	DCIC4	DCIC5	DCIC6	DCIC7		
	DCIC8	DCIC9	-etc.-						
"DVMREL1"	DMIM1	DMIM2	DMIM3	DMIM4	DMIM5	DMIM6	DMIM7		
	DMIM8	DMIM9	-etc.-						
"DVPREL2"	DPI2P1	DPI2P2	DPI2P3	DPI2P4	DPI2P5	DPI2P6	DPI2P7		
	DPI2P8	DPI2P9	-etc.-						
"DVCREL2"	DCI2C1	DCI2C2	DCI2C3	DCI2C4	DCI2C5	DCI2C6	DCI2C7		
	DCI2C8	DCI2C9	-etc.-						
"DVMREL2"	DMI2M1	DMI2M2	DMI2M3	DMI2M4	DMI2M5	DMI2M6	DMI2M7		
	DMI2M8	DMI2M9	-etc.-						
"DRESP2"	NRR1	NRR2	NRR3	NRR4	NRR5	NRR6	NRR7		
	NRR8	-etc.-							
"DVLREL1"	DLIL1	DLIL2	DLIL3	DLIL4	DLIL5	DLIL6	DLIL7		
	DLIL8	-etc.-							



Example:

DRESP2	1	LBUCK	5	3				
	DESVAR	101	3	4	5	1	205	209
		201						
	DTABLE	PI	YM	L				
	DRESP1	14	1	4	22	6	33	2
	DNODE	14	1	4	1	22	3	
		2	1	43	1			
	DVPREL1	101	102					
	DVCREL1	201	202					
	DVMREL1	301						
	DVPREL2	401	402					
	DVCREL2	501						
	DVMREL2	601	602	603				
	DRESP2	50	51					

Field	Contents
ID	Unique identification number. (Integer > 0)
LABEL	User-defined label. (Character)
EQID	DEQATN entry identification number. (Integer > 0)
FUNC	Function to be applied to the arguments. See Remark 8. (Character)
REGION	Region identifier for constraint screening. See Remark 5. (Integer > 0)
METHOD	When used with FUNC = BETA, METHOD = MIN indicates a minimization task while MAX indicates a maximization task. (Default = MIN) When used with FUNCT = MATCH, METHOD = LS indicated a least squares while METHOD = BETA indicated minimization of the maximum difference. (Default = LS)
Ci	Constants used when FUNC = BETA or FUNC = MATCH in combination with METHOD = BETA. See Remark 8. (Real, Defaults: C1 = 100., C2 = .005)
“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)
“DRESP1”	Flag indicating DRESP1 entry identification numbers. (Character)
NRk	DRESP1 entry identification number. (Integer > 0)



Field	Contents
“DNODE”	Flag indicating grid point and component identification numbers. (Character)
Gm	Identification number for any grid point in the model. (Integer > 0)
Cm	Component number of grid point Gm. (1 ≤ Integer ≤ 3)
DVPREL1	Flag indicating DVPREL1 entry identification number. (Character)
DPIPi	DVPREL1 entry identification number. (Integer > 0)
DVCREL1	Flag indicating DVCREL1 entry identification number. (Character)
DCICi	DVCREL1 entry identification number. (Integer > 0)
DVMREL1	Flag indicating DVMREL1 entry identification number. (Character)
DMIMi	DVMREL1 entry identification number. (Integer > 0)
DVPREL2	Flag indicating DVPREL2 entry identification number. (Character)
DPI2Pi	DVPREL2 entry identification number. (Integer > 0)
DVCREL2	Flag indicating DVCREL2 entry identification number. (Character)
DCI2Ci	DVCREL2 entry identification number. (Integer > 0)
DVMREL2	Flag indicating DVMREL2 entry identification number. (Character)
DMI2Mi	DVMREL2 entry identification number. (Integer > 0)
DRESP2	Flag indicating other DRESP2 entry identification number. (Character)
NRRk	DRESP2 entry identification number. (Integer > 0)
DVLREL1	Flag indicating other DVLREL1 entry identification number. (Character)
DLILI	DVLREL1 entry identification number. (Integer > 0)

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

A DRESP2 response can also be used as input on another DRESP2 entry or on an external response defined on DRESP3 entries.

DRESP2	ID	LABEL	EQID	REGION					
-	-	-	-	-					
“DTABLE”	LABL1	LABL2	-etc.-						
“DRESP2”	NRR1	NRR2	-etc.-						
-	-	-	-	-	-	-			



DRESP3	ID	LABEL	GROUP	TYPE	REGION			
	"DESVAR"	DVID1	DVID2	-etc.-				
-	-	-	-	-				
	"DRESP1"	NR1	NR2	-etc.-				
	"DRESP2"	NRR1	NRR2	-	-	-		

The DESOBJ Case Control command can also reference the DRESP2 entry ID:

$$\text{DESOBJ} \left(\begin{array}{c} \min \\ \max \end{array} \right) = n$$

Discussion:

It is often desirable to define responses that MSC Nastran does not calculate directly, e.g., local buckling criteria. Typically, 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 which 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, grid coordinate locations, first and second-level properties and other second level responses. The order of the arguments in the equation is dictated by the order in which they appear on the DRESP2. It is therefore expected that the number of arguments on the DEQATN will equal the number of DRESP2 arguments. The discussion of the [DEQATN](#) indicates that the number of DEQATN arguments may exceed those on the DRESP2. In this case, the missing arguments are implied frequencies.

The DRESP1 quantities called out on the DRESP2 can belong to separate superelements or subcases. If the DRESP1 quantities are not in the same subcase, it is necessary in Case Control to invoke the DRESP2 above the subcase level and to insert a DRSPAN command in the individual subcases to which the DRESP1 response belongs. DRESP2 quantities called out on a DRESP2 must belong in the same subcase.

An alternative to referencing an equation ID is to supply one of the available FUNC character strings:

Table 4-4

Function	Description
SUM	Sum of the arguments
AVG	Average of the arguments
SSQ	Sum of the squares of the arguments
RSS	Square root of the sum of the squares of the arguments
MAX	The largest value of the argument list
MIN	The smallest value of the argument list



Function	Description
BETA	Minimize the maximum response
MATCH	Match analysis results with user specified value

The meanings of the first six functions is the same as that given for the same functions described in [Table 4-2](#). The arguments used in the equations are all the arguments of the DRESP2 so that, for example, the AVG function can be used to find the average of number of design variable values.

The BETA function has been provided to simplify input when the user wishes to minimize the maximum response. The example in [Acoustic Optimization](#) discusses this technique and provides input fragments of doing this with and without the FUNC = BETA feature.

The DRESP2 with FUNC = BETA is intended to create the following design task

Minimize: $F(X_\beta) = C_1 X_\beta$

Subject to: $g = \frac{r_j - \gamma X_\beta}{C_3} \geq 0$

where C_1 and C_3 are user input values that have default values of 1.0 and 10.0, respectively. C_1 is used to scale the objective function and C_3 is used to offset the constraint bound from 0.

The γ quantity is computed so that the maximum constraint for all the response is equal to another user input value C_2 :

$$g_{max} = (r_{jmax} - \gamma X_\beta)/C_3 = c_2$$

where g_{max} is the value of the maximum response for the initial design. The default value for C_2 is 0.005, creating a maximum constraint that is just equal to the default value of DOPTPRM parameter GMAX.

The METHOD field on DRESP2 can be used in conjunction with FUNC = BETA to designate whether the task is to minimize the maximum response (METH = MIN) or maximize the minimum response (METH = MAX).

The MATCH function has been provided to simplify input when the user wishes to match analysis results with given values, such as those obtained in a test. Depending on the value of METHOD, one of two techniques is used to create the match.

When FUNC = MATCH and METHOD = LS, a least square approach is used.

The technique converts the user input into a response of the following form:

$$R_{match} = \sum_{j=1}^m \left(\frac{r_j - r_j^T}{r_j^T} \right)^2$$

where r_j is a response from a DRESP1 and r_j^T is a user defined target response.

Typically, this response would be designated as the objective function to be minimized but it could also be constrained.

When FUNC = BETA and METHOD = BETA, a beta approach is applied to minimize maximum deviation in a way similar to the FUNC = BETA method described above. In this case, the DRESP2 results in a design task to minimize a spawned design variable:

$$F(X_\beta) = C_1 X_\beta$$

subject to:

$$-\gamma X_\beta \leq \frac{(r_j - r_j^T)}{|r_j^T|} \leq \gamma X_\beta \quad j = 1, 2, \dots, m$$

Because X_β can become small, it is necessary to offset the constraint in a fashion similar to the BETA method given above.

Define:

$$r_j = \frac{(r_j - r_j^T)}{|r_j^T|}$$

and then determine R_{2max} and R_{2min} , the maximum and minimum values of r_j .

The γ quantity can then be determined from user specified values of C_2 and C_3 using the following equation:

$$\frac{Max(R_{2max}, R_{2min}) - \gamma X_\beta}{C_3} = C_2$$

In this case, the DRESP2 can only be invoked as the objective and cannot be constrained.

DRESP3

Purpose:

Defines an external response using user-supplied routines. The response can be used as a constraint or as an objective.

Entry Description:

1	2	3	4	5	6	7	8	9	10
DRESP3	ID	LABEL	GROUP	TYPE	REGION				
	"DESVAR"	DVID1	DVID2	DVID3	DVID4	DVID5	DVID6	DVID7	



		DVID8	-etc.-						
	"DTABLE"	LABL1	LABL2	LABL3	LABL4	LABL5	LABL6	LABL7	
		LABL8	-etc.-						
	"DRESP1"	NR1	NR2	NR3	NR4	NR5	NR6	NR7	
		NR8	-etc.-						
	"DNODE"	G1	C1	G2	C2	G3	C3		
		G4	C4	etc.					
	"DVPREL1"	DPIP1	DPIP2	DPIP3	DPIP4	DPIP5	DPIP6	DPIP7	
		DPIP8	DPIP9	-etc.-					
	"DVCREL1"	DCIC1	DCIC2	DCIC3	DCIC4	DCIC5	DCIC6	DCIC7	
		DCIC8	DCIC9	-etc.-					
	"DVMREL1"	DMIM1	DMIM2	DMIM3	DMIM4	DMIM5	DMIM6	DMIM7	
		DMIM8	DMIM9	-etc.-					
	"DVPREL2"	DPI2P1	DPI2P2	DPI2P3	DPI2P4	DPI2P5	DPI2P6	DPI2P7	
		DPI2P8	DPI2P9	-etc.-					
	"DVCREL2"	DCI2C1	DCI2C2	DCI2C3	DCI2C4	DCI2C5	DCI2C6	DCI2C7	
		DCI2C8	DCI2C9	-etc.-					
	"DVMREL2"	DMI2M1	DMI2M2	DMI2M3	DMI2M4	DMI2M5	DMI2M6	DMI2M7	
		DMI2M8	DMI2M9	-etc.-					
	"DRESP2"	NRR1	NRR2	NRR3	NRR4	NRR5	NRR6	NRR7	
		NRR8	-etc.-						
	"DVLREL1"	DLIL1	DLIL2	DLIL3	DLIL4	DLIL5	DLIL6	DLIL7	
		DLIL8	-etc.-						
	"USRDATA"	Character String Data							
		-etc.-							

Example:

DRESP3	1	LBUCK	TAILWNG	BUCK					
	DESVAR	101	3	4	5	1	205	209	
		201							
	DTABLE	PI	YM	L					
	DRESP1	14	1	4	22	6	33	2	
	DNODE	14	1	4	1	22	3		
		2	1	43	1				
	DVPREL1	101	102						
	DVCREL1	201	202						
	DVMREL1	301							
	DVPREL2	401	402						
	DVCREL2	501							



	DVMREL2	601	602	603					
	DRESP2	50	51						
	USRDATA	1234 Material Constants							

Field	Contents
ID	Unique identification number. (Integer > 0)
LABEL	User-defined label. (Character)
GROUP	Group of external response that this type belongs to. (Character)
TYPE	External response type. (Character)
"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)
"DRESP1"	Flag indicating DRESP1 entry identification numbers. (Character)
NRk	DRESP1 entry identification number. (Integer > 0)
"DNODE"	Flag signifying that the following fields are grid points.
Gm	Grid point identification number. (Integer > 0)
Cm	Degree-of-freedom number of grid point GM. ($1 \leq \text{Integer} \leq 3$)
DVPREL1	Flag indicating DVPREL1 entry identification number. (Character)
DPIPi	DVPREL1 entry identification number. (Integer > 0)
DVCREL1	Flag indicating DVCREL1 entry identification number. (Character)
DCICi	DVCREL1 entry identification number. (Integer > 0)
DVMREL1	Flag indicating DVMREL1 entry identification number. (Character)
DMIMi	DVMREL1 entry identification number. (Integer > 0)
DVMPREL2	Flag indicating DVMPREL2 entry identification number. (Character)
DPI2Pi	DVPREL2 entry identification number. (Integer > 0)
DVCREL2	Flag indicating DVCREL2 entry identification number. (Character)
DCI2Ci	DVCREL2 entry identification number. (Integer > 0)
DVMREL2	Flag indicating DVMREL2 entry identification number. (Character)
DMI2Mi	DVMREL2 entry identification number. (Integer > 0)
DRESP2	Flag indicating other DRESP2 entry identification number. (Character)
NRRk	DRESP2 entry identification number. (Integer > 0)
DVLREL1	Flag indicating other DVLREL1 entry identification number. (Character)



Field	Contents
DLILI	DVLREL1 entry identification number. (Integer > 0)
USRDATA	Flag indicating user input data. (Character)



Associated Entries:

The DCONSTR entry can be used to place bounds on a DRESP3 response using the DRESP3 entry ID as a reference:

DCONSTR	DCID	RID	LALLOW	UALLOW				
---------	------	-----	--------	--------	--	--	--	--

The DESOBJ Case Control command can also reference the DRESP3 entry ID.

$$\text{DESOBJ} \begin{pmatrix} \text{min} \\ \text{max} \end{pmatrix} = n$$

Discussion:

As discussed in [Design Responses](#), the DRESP3 entry enables the consideration of responses that are not direct MSC Nastran quantities and that cannot be formulated from MSC Nastran quantities through the use of the DRESP2 entry. Instead, the DRESP3 invokes an API (application programming interface) to compute the response using a combination of MSC Nastran supplied data and user supplied data and algorithms. It is seen that the input format for the DRESP3 is very similar to the DRESP2 in that it permits a variety of types of information to be used as arguments. The DRESP3 does not use the DEQATN, so there is no EQID identified. Instead, the GROUP name refers to an FMS CONNECT entry (See [File Management](#)) that defines the API evaluator while the TYPE argument identifies the particular response type that is to be used to evaluate the response. Another difference from the DRESP2 is the provision for another type of data designed USRDATA. This character string data provides a convenient way of communicating to the API and can include, for example, a file name that needs to be opened or some special instructions. The server program parses this character string and takes the appropriate action. As with the DRESP2, it is necessary to use the DRSPAN Case Control command when the DRESP1 quantities come from more than one subcase.

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, DRESP2 or DRESP3 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.-					

DRESP3	ID	LABEL	GROUP	TYPE	REGION				
	"DESVAR"	DVID1	DVID2	-etc.-					
	-	-	-	-					
	"DRESP1"	NR1	NR2	-etc.-					
	"DRESP2"	NRR1	NRR2						

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. One application where it has utility is when you are performing a sensitivity analysis and want to make sure that the sensitivities of all the responses of interest are provided. The TRS value can then be set to a large negative number (say -1000.) and the NSTR value can be set to a number equal to or greater than the number of responses.

The RTYPE refers to the RTYPE used on the DRESP1. For the DRESP2's, the applicable RTYPE is EQUA. For the DRESP3's, the applicable RTYPE for constraint screening is DRESP3. 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 MSC 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.

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 in [Table 4-3](#) as a function of Response type. NSTR, or the number of constraints to be retained per region, has a default value of 20, which again can be overridden. The NSTR value can not be used for response types WEIGHT, VOLUME, EIGN, LAMA, CEIG, FRMASS, COMP and TOTSE.



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 or Integer)

Associated Entries:

DRESP2, DRESP3, SEDRSP2, SEDRSP3 and DVPREL2, DVCREL2, DVMREL2 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.-					
	-	-	-	-	-	-			

DRESP3	ID	LABEL	GROUP	TYPE	REGION				
	"DESVAR"	DVID1	DVID2	-etc.-					
	"DTABLE"	LABL1	LABL2	-etc.-					
	-	-	-	-					

DVPREL2	ID	TYPE	PID	FID	PMIN	PMAX	EQID		
	"DESVAR"	DVID1	DVID2	DVID3	-etc.-				
	"DTABLE"	LABL1	LABL2	LABL3	-etc.-				

DVCREL2	ID	TYPE	EID	CPNAME	CPMIN	CPMAX	EQID		
	"DESVAR"	DVID1	DVID2	DVID3	-etc.-				
	"DTABLE"	LABL1	LABL2	LABL3	-etc.-				



DVMREL2	ID	TYPE	MID	MINAME	MPMIN	MPMAX	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 in 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 “DTABLE” argument list is defined on a DVPREL2 or other entry, the constant is referenced by its name.

Multiple DTABLE entries may appear in the Bulk Data.

For PART SE, if LABLi is referenced on SEDRSP2 and/or SEDRSP3, DTABLE entries must be placed in a PART SE where companion design model entries, such as DESVAR, DRESP1 and etc, are available.

If the VALUi is an integer, this points to the ID of a TABLEDx Bulk Data entry that lists the constants as a function of frequency or time.

DTABLE2

Purpose

Defines real constants from a field of property, material or connections bulk data entries which then can be used with DEQATN equations.



Entry Description:

1	2	3	4	5	6	7	8	9	10
DTABLE2	LABL1	PNAME1	PID1	FNAME1	LABL2	PNAME2	PID2	FNAME2	
	LABL3	PNAME3	PID3	FNAME3					

Field Contents

LABLi	Label for the constant. (Character)
PNAMEi	Property, material or connection bulk data entry name. (Character)
PIDI	ID of PNAMEi entry. (Integer > 0)
FNAMEi	Field name of PNAMEi. (Character)

Discussion:

LABLi on DTABLE2 and DTABLE must be unique. LABLi on DTABLE2 can be referenced under DTABLE flag of DVxREL2 (where x=P, M or C)/DRESP2/DRESP3/SEDRSP2/SEDRSP3. Values for the FNAMEi field of the PNAMEi Bulk Data entry with the ID of PIDi are taken from analysis model before updating of analysis values with the designed value. If the updated value is desired, use the DVxREL2 flag on DRESP2 or DRESP3 entries instead. FNAMEi must be the same as the character string that appears on the PNAMEi Bulk Data entry.

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:

Design variables must first be defined using DESVAR entries.

DESVAR	ID	LABEL	XINIT	XLB	XUB	DELXV			
--------	----	-------	-------	-----	-----	-------	--	--	--

Discussion:

In shape optimization, shape basis vectors relate the changes in a design variable to changes in grid locations. Analytical Boundary Shapes are calculated in an auxiliary model as described in [Relating Design Variables to Shape Changes](#). The DVBSHAP entry is then used to identify the auxiliary model and then apply scale factors to a series of solution vectors. If multiple DVBSHAP entries share the same design variable ID, the shapes are added.

DVCREL1

Purpose:

Defines the relation between a connectivity property and design variables.



Entry Description:

DVCREL1	ID	TYPE	EID	CFNAME	CPMIN	CPMAX	C0		
	DVID1	COEF1/ PVAL	DVID2	COEF2	DVID3	COEF3	-etc.-		

Field	Contents
ID	Unique identification number. (Integer > 0)
TYPE	Name of an element connectivity entry, such as "CBAR", "CQUAD", etc. (Character)
EID	Element Identification number. (Integer > 0)
CPNAME	Name of connectivity property, such as "X1", "X2", "X3", "ZOFFS", etc. (Character)
CPMIN	Minimum value allowed for this property. If CPNAME references a connectivity property that can only be positive, then the default value of CPMIN is 1.0E-20. Otherwise, it is -1.0E35. (Real)
CPMAX	Maximum value allowed for this property. (Real; Default = 1.0E+20)
C0	Constant term of relation. (Real; Default = 0.0)
DVIDi	DESVAR entry identification number. (Integer > 0)
COEFi	Coefficient of linear relation. (Real)
'PVAL'	Flag to indicate COEF2 is to be set to the current connectivity property. (Character = PVAL)

Associated Entries:

Design variables are referenced on DVCREL1 entries by their DESVAR-defined IDs.

DESVAR	ID	LABEL	XINIT	XLB	XUB	DELXV			
--------	----	-------	-------	-----	-----	-------	--	--	--



The DVCREL1 can be included in DRESP2 or DRESP3 entries:

DRESP2	ID	LABEL	EQID	REGION					
	“DESVAR”	DVID1	DVID2	-etc.-					
-	-	-	-	-					
	“DRESP1”	NR1	NR2	-etc.-					
-	-	-	-	-	-	-			
	“DVCREL1”	DCIC1	DCIC2	-etc.-					

DRESP3	ID	LABEL	GROUP	TYPE	REGION				
	“DESVAR”	DVID1	DVID2	-etc.-					
-	-	-	-	-					
-	-	-	-	-					
-	-	-	-	-	-	-			
	“DVCREL1”	DCIC1	DCIC2	-etc.-					

Discussion:

A “connectivity” property is one that has been defined on a Bulk Data entry that begins with the letter “C”. The design of these properties enables you to design things such as beam offsets and concentrated masses. A connectivity property can be expressed as a linear combination of design variables as

$$p_1 = c_0 + c_1x_1 + c_2x_2 + \dots + c_nx_n$$

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.

The CPNAME is a character string that identifies the required property value by using the same name as that given in the [MSC Nastran Quick Reference Guide](#). [Table 2-3](#) of this User’s Guide identifies the properties that are available for design.

It is seen that limits can be placed on the property by specifying CPMIN and CPMAX values. This is in addition to the limits that are provided by the XLB/XUB fields on the DESVAR entries. If possible, the limits on the DESVAR entries should be more restrictive than those on the DVCREL1 entries. This is because the optimizer will never allow a design variable limit to be violated, but it may allow a property limit to be violated if this seems to help satisfy some other violated constraint. Violating a connectivity constraint could have disastrous results if it produces an element that is physically meaningless.

It is seen that there is a redundancy in the computation of the connectivity property value in that it is the product of the user defined coefficients and the design variable value. A guideline is that the coefficients should be specified in a way that allows the design variable values to be near unity. The optimizer performs best when all the design variables are in the same range and, in particular, when they are all on the order of 1.0.



If the term ‘PVAL’ is entered in the COEF1 field, this is a flag that the COEF1 value is to be obtained from the corresponding property on the connectivity entry. This simplifies the process of completing this entry when the desire is to have a one-to-one correspondence between the designed variable and the designed property. The assumption is that the DESVAR entry identified by DVID1 is 1.0.

DVCREL2

Purpose:

Defines a connectivity property by reference to an equation defined on a DEQATN entry.

Entry Description:

DVCREL2	ID	TYPE	EID	CPNAME	CPMIN	CPMAX	EQID		
	“DESVAR”	DVID1	DVID2	-etc.-					
	“DTABLE”	LABI1	LABL2	-etc.-					

Field	Contents
ID	Unique identification number. (Integer > 0)
TYPE	Name of an element connectivity entry, such as “CBAR”, “CBEAM”, etc. (Character)
EID	Element Identification number. (Integer > 0)
CPNAME	Connectivity property name, such as “Z1”, “ZOFFS”, etc. (Character)
CPMIN	Minimum value allowed for this property. If CPNAME references a connectivity property that can only be positive, then the default value of CPMIN is 1.0E-20. Otherwise, it is -1.0E35. (Real)
CPMAX	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 or DTABLE2 entry follow. This field may be omitted if there are no constants involved in this relation. (Character)
LABi	Label for constant on the DTABLE or DTABLE2 entry. (Integer > 0)



Associated Entries:

To define a synthetic property relation, the DVCREL2 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			
DTABLE	LBL1	VALU1	LBL2	VALU2	LBL3	VALU3	LBL4	VALU4	
	LBL5	VALU5	LBL6	VALU6	-etc.-				

The DVCREL2 entries can be included in DRESP2 or DRESP3 entries.

DRESP2	ID	LABEL	EQID	REGION					
	“DESVAR”	DVID1	DVID2	-etc.-					
	-	-	-	-					
	“DRESP1”	NR1	NR2	-etc.-					
	-	-	-	-	-	-			
	“DVCREL2”	DCIC1	DCIC2	-etc.-					

DRESP3	ID	LABEL	GROUP	TYPE	REGION				
	“DESVAR”	DVID1	DVID2	-etc.-					
	-	-	-	-					
	-	-	-	-					
	-	-	-	-	-	-			
	“DVCREL2”	DCIC1	DCIC2	-etc.-					

Discussion:

For some applications, the linear design variable-to-property relations provided by the DVCREL1 entry may not be sufficient. In these cases, equations may be defined to relate design variables to properties in a nonlinear fashion. The DVCREL2 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 or DTABLE2 entry.

Every property defined on a DVCREL2 entry becomes an independently varying property internally in MSC 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. Care should be taken to not use the DVCREL2 entry when the relationship between the design variable and the connectivity property is linear; the DVCREL1 entry should be used in this case. This is because



the DVCREL2 entry requires the added design sensitivity calculations and could therefore impact the performance.

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 $Ni \neq 0.0$)

Associated Entries:

The DESVAR entry defines a design variable that can be used to describe grid variations as well as other design relations. The GRID entry (not shown) provides the baseline locations for the designed grid.

DESVAR	ID	LABEL	XINIT	XLB	XUB	DELXV			
--------	----	-------	-------	-----	-----	-------	--	--	--

Discussion:

Changes in grid coordinate values are expressed as functions of design variables according to the relation

$$\{\Delta G_i\} = \sum_j \{T\}_{ij} \cdot \Delta x_j$$

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 [Basis Vectors in Shape Optimization](#).

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 result in vectorial addition of the corresponding $\{T\}_{ij}$.

Other shape basis vector definition methods utilize DVBSHAP and DVSHAP Bulk Data entries.



DVLREL1

Purpose:

Defines the linear relation between analysis model loading and design variables.

Entry Description:

DVLREL1	ID	TYPE	SID	LNAME	LMIN	LMAX	C0			
	ATT1	ATT2	ATT3	ATT4	ATT5					
	DVID	COEF	DVID2	COEF2	DVID3	Etc				

Field	Contents
ID	Unique identification number (Integer>0)
TYPE	Name of Load, such as FORCE. (Character)
SID	Load set ID (Integer>0)
LNAME	Load Name, such as F or N1 on the FORCE entry. (Character)
LMIN	Minimum value for the load. (Real, default=-1.0e35)
LMAX	Maximum value for the load (Real, Default=1.0e20)
C 0	Constant term of relation (Real, Default=0.0)
ATTi	Attributes of the designed load, see Table 4.5 (Integer>0 or blank)
DVIDi	DESVAR entry identification number. (Integer>0)
COEFi	Coefficient of linear relation or keyword="PVAL", (If i=1, Real or Character; if i>1, Real)

Associated Entries:

Design variables are referenced on DVLREL1 entries by their DESVAR-defined IDs.

DESVAR	ID	LABEL	XINIT	XLB	XUB	DELXV				

The DVLREL1 can be included in DRESP2 or DRESP3 entries:

DRESP2	ID	LABEL	EQID	REGION					
	“DESVAR”	DVID1	DVID2	-etc.-					
-	-	-	-	-					
	“DRESP1”	NR1	NR2	-etc.-					
-	-	-	-	-	-	-			
	“DVLREL1”	DLIL1	DLIL2	-etc.-					

DRESP3	ID	LABEL	GROUP	TYPE	REGION				
	“DESVAR”	DVID1	DVID2	-etc.-					
-	-	-	-	-					
-	-	-	-	-					
-	-	-	-	-	-	-			
	“DVLREL1”	DLIL1	DLIL2	-etc.-					

Discussion:

A “load” property is one that has been defined on a FORCE, MOMEMT or LOAD Bulk Data entry. The design of these properties enables you to investigate the effect an external load has on designated responses in a Statics Analysis. It is noted that this design quantity is limited in the applied loads entries that it can design and that is it is limited to subcases with ANALYSIS=STATICS. The primary application of the DVLREL1 at this point is felt to be in sensitivity analysis. A particular instance is where a global/local analysis is being performed and you can now determine how the internal loads from the global analysis drive responses in a local analysis when they become applied loads in the local analysis. A load property can be expressed as a linear combination of design variables as

$$p_1 = c_0 + c_1x_1 + c_2x_2 + \dots + c_nx_n$$

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.

The LNAME is a character string that identifies the required property value by using the same name as that given in the [MSC Nastran Quick Reference Guide](#). Table 2-4 of this User's Guide identifies the properties that are available for design.

It is seen that limits can be placed on the property by specifying LPMIN and LPMAX values. This is in addition to the limits that are provided by the XLB/XUB fields on the DESVAR entries. If possible, the limits on the DESVAR entries should be more restrictive than those on the DVLREL1 entries. This is because the optimizer will never allow a design variable limit to be violated, but it may allow a load limit to be violated if this seems to help satisfy some other violated constraint.

It is seen that there is a redundancy in the computation of the load property value in that it is the product of the user defined coefficients and the design variable value. A guideline is that the coefficients should



be specified in a way that allows the design variable values to be near unity. The optimizer performs best when all the design variables are in the same range and, in particular, when they are all on the order of 1.0.

If the term ‘PVAL’ is entered in the COEF1 field, this is a flag that the COEF1 value is to be obtained from the corresponding property on the load entry. This simplifies the process of completing this entry when the desire is to have a one-to-one correspondence between the designed variable and the designed load. The assumption is that the DESVAR entry identified by DVID1 is 1.0.

DVMREL1

Purpose:

Defines the relation between a material property and design variables.

Entry Description:

DVMREL1	ID	TYPE	MID	MPNAME	MPMIN	MPMAX	C0		
	DVID1	COEF1/ 'PVAL'	DVID2	COEF2	DVID3	COEF3	-etc.-		

Field	Contents
ID	Unique identification number. (Integer > 0)
TYPE	Name of a material property entry, such as “MAT1”, “MAT2”, etc. (Character)
MID	Material Identification number. (Integer > 0)
MPNAME	Name of material property, such as “E”, “RHO”, etc. (Character)
MPMIN	Minimum value allowed for this property. If MPNAME references a material property that can only be positive, then the default value of MPMIN is 1.0E-20. Otherwise, it is -1.0E35. (Real)
MPMAX	Maximum value allowed for this property. (Real; Default = 1.0E+20)
C0	Constant term of relation. (Real; Default = 0.0)
DVIDi	DESVAR entry identification number. (Integer > 0)
COEFi	Coefficient of linear relation. (Real)
‘PVAL’	Flag to indicate COEF1 is to be set to the current material property. (Character = ‘PVAL’)

Associated Entries:

Design variables are referenced on DVMREL1 entries by their DESVAR-defined IDs.



DESVAR	ID	LABEL	XINIT	XLB	XUB	DELXV			
--------	----	-------	-------	-----	-----	-------	--	--	--

The DVMREL1 can be included in DRESP2 or DRESP3 entries:

DRESP2	ID	LABEL	EQID	REGION					
	“DESVAR”	DVID1	DVID2	-etc.-					
-	-	-	-	-					
	“DRESP1”	NR1	NR2	-etc.-					
-	-	-	-	-	-	-			
	“DVMREL1”	DMIM1	DMIM2	-etc.-					

DRESP3	ID	LABEL	GROUP	TYPE	REGION				
	“DESVAR”	DVID1	DVID2	-etc.-					
-	-	-	-	-	-	-			
	“DVMREL1”	DMIM1	DMIM2	-etc.-					

Discussion:

A material property is a real number that has been defined on a MATi Bulk Data entry. The design of these properties allows you to vary such key parameters as Young's Modulus and material density. A material property can be expressed as a linear combination of design variables as

$$p_1 = c_0 + c_1x_1 + c_2x_2 + \dots + c_nx_n$$

Both independent as well as dependent design variables can appear in this relation (see the DLINK entry). This form 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.

The MPNAME is a character string that identifies the required property value by using the same name as that given in the *MSC Nastran Quick Reference Guide*. Table 2-2 of this User's Guide identifies the properties that are available for design.

It is seen that limits can be placed on the property by specifying MPMIN and MPMAX values. This is in addition to the limits that are provided by the XLB/XUB fields on the DESVAR entries. If possible, the limits on the DESVAR entries should be more restrictive than those on the DVMREL1 entries. This is because the optimizer will never allow a design variable limit to be violated, but it may allow a property limit to be violated if this seems to help satisfy some other violated constraint. Violating a material property constraint could have disastrous results if it produces a material that is physically meaningless.

It is seen that there is a redundancy in the computation of the material property value in that it is the product of the user defined coefficients and the design variable value. A guideline is that the coefficients should be specified in a way that allows the design variable values to be near unity. The optimizer performs best when all the design variables are in the same range and, in particular, when they are all on the order of 1.0.



If the term ‘PVAL’ is entered in the COEF1 field, this is a flag that the COEF1 value is to be obtained from the corresponding property on the material entry. This simplifies the process of completing this entry when the desire is to have a one-to-one correspondence between the designed variable and the designed property. The assumption is that the DESVAR entry identified by DVID1 is 1.0.



DVMREL2

Purpose:

Defines a connectivity property by reference to an equation defined on a DEQATN entry.

Entry Description:

DVMREL2	ID	TYPE	EID	MPNAME	MPMIN	MPMAX	EQID		
	“DESVAR”	DVID1	DVID2	-etc.-					
	“DTABLE”	LABL1	LABL2	-etc.-					

Field	Contents
ID	Unique identification number. (Integer > 0)
TYPE	Name of a material property entry, such as “MAT1”, “MAT2”, etc. (Character)
MID	Material Identification number. (Integer > 0)
MPNAME	Name of material property, such as “E”, “RHO”, etc. (Character)
MPMIN	Minimum value allowed for this property. If MPNAME references a material property that can only be positive, then the default value of MPMIN is 1.0E-20. Otherwise, it is -1.0E35. (Real)
MPMAX	Maximum value allowed for this property. (Real; Default = 1.0E+20)
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 or DTABLE2 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 or DTABLE2 entry. (Integer > 0)

Associated Entries:

To define a synthetic property relation, the DVMREL2 entry identifies a DEQATN entry and declares the design variable (DESVAR) and any necessary table constant (DTABLE, DTABLE2) arguments.

DEQATN	EQID	EQUATION			
		EQUATION (Cont.)			

DESVAR	ID	LABEL	XINIT	XLB	XUB	DELXV			
--------	----	-------	-------	-----	-----	-------	--	--	--



DTABLE	LABL1	VALU1	LABL2	VALU2	LABL3	VALU3	LABL4	VALU4	
	LABL5	VALU5	LABL6	VALU6	-etc.-				

The DVMREL2 entries can be included in DRESP2 or DRESP3 entries.

DRESP2	ID	LABEL	EQID	REGION					
	“DESVAR”	DVID1	DVID2	-etc.-					
-	-	-	-	-					
	“DRESP1”	NR1	NR2	-etc.-					
-	-	-	-	-	-	-			
	“DVMREL2”	DMI2M1	DMI2M2	-etc.-					

DRESP3	ID	LABEL	GROUP	TYPE	REGION				
	“DESVAR”	DVID1	DVID2	-etc.-					
-	-	-	-	-					
-	-	-	-	-	-	-			
	“DVMREL2”	DMI2M1	DMI2M2	-etc.-					

Discussion:

For some applications, the linear design variable-to-property relations provided by the DVMREL1 entry may not be sufficient. In these cases, equations may be defined to relate design variables to material properties in a nonlinear fashion. The DVMREL2 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 or DTABLE2 entry.

Every property defined on a DVMREL2 entry becomes an independently varying property internally in MSC 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. Care should be taken to not use the DVMREL2 entry when the relationship between the design variable and the material property is linear; the DVMREL1 entry should be used in this case. This is because the DVMREL2 entry requires the added design sensitivity calculations and could therefore impact the performance.



DVPREL1

Purpose:

Defines the relation between an analysis model property and design variables.

Format:

1	2	3	4	5	6	7	8	9	10
DVPREL1	ID	TYPE	PID	PNAME/ FID	PMIN	PMAX	C0		
	DVID1	COEF1/ 'PVAL'	DVID2	COEF2	DVID3	-etc.-			

Example:

DVPREL1	12	PBAR	612	6	0.2	3.0			
	4	0.25	20	20.0	5	0.3			

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)
PNAME/FID	Property name, such as "T", "A", or field position of the property entry, or word position in the element property table of the analysis model. (Character or Integer ≠ 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 = 1.0E-20)
PMAX	Maximum value allowed for this property. (Real; Default = 1.0E+20)
C0	Constant term of relation. (Real; Default = 0.0)
DVIDi	DESVAR entry identification number. (Integer > 0)
COEFi	Coefficient of linear relation. (Real)
'PVAL'	Flag to indicate COEF1 is to be set to the current property value (Character = 'PVAL')

Associated Entries:

Design variables are referenced on DVPREL1 entries by their DESVAR-defined IDs.

DESVAR	ID	LABEL	XINIT	XLB	XUB	DELVXV			
--------	----	-------	-------	-----	-----	--------	--	--	--



The DVPREL1 IDs can be included in DRESP2 or DRESP3 entries:

DRESP2	ID	LABEL	EQID	REGION					
	“DESVAR”	DVID1	DVID2	-etc.-					
-	-	-	-	-					
	“DRESP1”	NR1	NR2	-etc.-					
-	-	-	-	-	-	-			
	“DVPREL1”	DPIP1	DPIP2	-etc.-					

DRESP3	ID	LABEL	GROUP	TYPE	REGION				
	“DESVAR”	DVID1	DVID2	-etc.-					
-	-	-	-	-					
-	-	-	-	-	-	-			
	“DVPREL1”	DPIP1	DPIP2	-etc.-					

Discussion:

An analysis model property can be expressed as a linear combination of design variables as

$$p_1 = c_0 + c_1x_1 + c_2x_2 + \dots + c_nx_n$$

Both independent as well as dependent design variables can appear in this relation (see the DLINK entry). This form 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.

The TYPE typically identifies the property type; e.g., PBAR, being designed with TYPE=GPLY a special case that allows for the design of a global ply id that can span multiple PCOMPG entries.

The PNAME is a character string that identifies the required property value by using the same name as that given in the [MSC Nastran Quick Reference Guide](#). Table 2-1 of this User’s Guide identifies the properties that are available for design.

Limits can be placed on the property by specifying PMIN and PMAX values. This is in addition to the limits that are provided by the XLB/XUB fields on the DESVAR entries. If possible, the limits on the DESVAR entries should be more restrictive than those on the DVPREL1 entries. This is because the optimizer will never allow a design variable limit to be violated, but it may allow a property limit to be violated if this seems to help satisfy some other violated constraint. Violating a material property constraint could have disastrous results if it produces a material that is physically meaningless.

Note that there is a redundancy in the computation of the property value in that it is the product of the user defined coefficients and the design variable value. A guideline is that the coefficients should be specified in a way that allows the design variable values to be near unity. The optimizer performs best when all the design variables are in the same range and, in particular, when they are all on the order of 1.0.



If the term ‘PVAL’ is entered in the COEF1 field, this is a flag that the COEF1 value is to be obtained from the corresponding property entry. This simplifies the process of completing this entry when the desire is to have a one-to-one correspondence between the designed variable and the designed property. The assumption is that the DESVAR entry identified by DVID1 is 1.0.

Guidelines for the PCOMP and PCOMPG Entries

[Table 2-1](#) indicates that all the real values on the PCOMP and PCOMPG entries can be designed. Two of these merit further discussion

The ZO property indicates the offset from the reference plane to the bottom surface. If this is left blank, the value defaults to half the total thickness. It is not necessary to design the ZO quantity if the user input value is blank and the half thickness value is acceptable since the ZO value adjusts to half the total thickness in this case.

The ORIENT*i* field is a difficult property to design since the composite materials vary as the fourth power of the quantity. Suggested guidelines are to:

- Limit the allowable movement of the designed orientation to no more than $\pm 20^\circ$.
- Avoid design variable values of 0.0 because it is difficult to move away from this value. It is better to apply an offset such as $\theta_i = -1 + X_i$ so that an X_i of 1.0 gives a zero orientation angle.

A special situation occurs for the PCOMPG where it is likely that the user wants to design the thickness or orientation of a ply id that spans multiple PCOMPG entries. In this case, as discussed above, the TYPE is set to GPLY. Then the PID is set to the PLY ID value on the PCOMPG’s and the PNAME can be either T or THETA. In this situation, the relationship between the global ply value and the design variables is given by:

$$P_i = C0 + (T0_i \text{ or } THETA0_i) \cdot \sum_j (COEF_j \cdot X_{DVID_j}) \text{ for PNAME=T or THETA}$$

Use of the C0 term is not recommended in this situation and it is seen the designed property uses the initial thickness or theta value in its computation. This is because a global ply can have different thicknesses on different PCOMPG entries. If THETA0 is initially zero, it is reset to 1.0 in this case.

DVPREL2

Purpose:

Defines a structural property by reference to an equation defined on a DEQATN entry.



Entry Description:

DVPREL2	ID	TYPE	PID	FNAME	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)
PNAME/FID	Property name, such as “T”, “A”, or field position of the property entry, or word position in the element property table of the analysis model. (Character or Integer ≠ 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 = 1.E-20)
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 LABLs for the constants in a DTABLE or DTABLE2 entry follow. This field may be omitted if there are no constants involved in this relation. (Character)
LABLi	Label for a constant on the DTABLE or DTABLE2 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, DTABLE2) arguments.



DEQATN	EQID	EQUATION					
		EQUATION (Cont.)					

DESVAR	ID	LABEL	XINIT	XLB	XUB	DELXV			
--------	----	-------	-------	-----	-----	-------	--	--	--

DTABLE	LABL1	VALU1	LABL2	VALU2	LABL3	VALU3	LABL4	VALU4	
	LABL5	VALU5	LABL6	VALU6	-etc.-				

The DVPREL2 entries can be included in DRESP2 or DRESP3 entries.

DRESP2	ID	LABEL	EQID	REGION					
	“DESVAR”	DVID1	DVID2	-etc.-					
-	-	-	-						
	“DRESP1”	NR1	NR2	-etc.-					
-	-	-	-	-	-	-			
	DVPREL2	DP2P1	DP2P2	-etc.-					

DRESP3	ID	LABEL	GROUP	TYPE	REGION				
	“DESVAR”	DVID1	DVID2	-etc.-					
-	-	-	-						
-	-	-	-	-	-	-			
	DVPREL2	DP2P1	DP2P2	-etc.-					

Discussion:

For some 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.

Every property defined on a DVPREL2 entry becomes an independently varying property internally in MSC 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. Care should be taken to not use the DVPREL2 entry when the relationship between the design variable and the property is linear; the DVPREL1 entry should be used in this case. This is because the DVPREL2 entry requires the added design sensitivity calculations and could therefore impact the performance.



DVPSURF

Purpose:

Defines the relationship between a control surface setting in a particular subcase and a design variable.

Entry Description:

DVPSURF	ID	AELABEL	TRIMID	DVID	COEF				
---------	----	---------	--------	------	------	--	--	--	--

Field	Contents
ID	Unique identification number (Integer>0)
AELABEL	LABEL of the AESURF entry that is being designed (Character ,no default)
TRIMID	Associated trim set identification number (Integer>0)
DVID	DESVAR entry identification number (Integer>0)
COEF	Coefficient of linear relation (Real)

The DVPSURF is a special purpose entry that is limited to static aeroelasticity (ANALYSIS=SAERO). It provides a linear relationship between a control surface setting and a design variable:

$$\delta_{\text{SURF}} = \text{COEF} * X_{\text{DVID}}$$

The AELABEL points to a particular AESURF entry that is being designed while the TRIMID points to the subcase in which the user wants to design the control surface setting. This enables finding the values of control surface settings during a maneuver that results, for example in the minimum root bending.

DVSHAP

Purpose:

Defines a shape basis vector by relating a design variable identification number (DVID) to columns of a displacement matrix.



Format:

1	2	3	4	5	6	7	8	9	10
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. See Remark 2. ($1 \leq \text{Integer} \leq$ 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:

The design variables must first be defined using DESVAR entries.

DESVAR	ID	LABEL	XINIT	XLB	XUB	DELXV			
--------	----	-------	-------	-----	-----	-------	--	--	--

The displacement vector are DBLOCATE'd using File Management Statements as shown in [File Management](#).

Discussion:

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 DVSHAP entry is applied to solution vectors that are output in a separate analysis. The DVSHAP entry is used to apply scale factors to a series of solution vectors. If multiple DVSHAP entries share the same design variable ID, the shapes are added.

MODTRAK

Purpose:

Specifies parameters for mode tracking in design optimization (SOL 200).



Format:

1	2	3	4	5	6	7	8	9	10
MODTRAK	SID	LOWRNG	HIGHRNG	MTFILTER					

Field	Contents
SID	Sets identification number that is selected in the Case Control Section with the MODTRAK command. (Integer; No Default)
LOWRNG	Lowest mode number in range to search. (Integer ≥ 0 , Default = 0. If nonzero, LOWRNG < HIGHRNG.)
HIGHRNG	Highest mode number in range to search. (Integer > 0 , Default = number of eigenvalues extracted. If nonzero, LOWRNG < HIGHRNG.)
MTFILTER	Filtering parameter used in mode cross-orthogonality check. (Real, Default = 0.9)

Associated Entries:

The MODTRAK entry is invoked by the Case Control command:

MODTRAK = n

Discussion:

The concept of mode tracking is presented in [Multidisciplinary Analysis](#). Mode tracking is available only in normal modes analysis (ANALYSIS = MODES). The MODTRAK Bulk Data entry specifies a range of modes (by sequence number) that are to be tracked and a threshold value MTFILTER that is used to designate whether modes correlate. The default value of 0.9 is applied to normalized results of the triple matrix product of [Equation \(2-22\)](#) and is usually adequate.

SEDLINK

Purpose:

Relates one design variable of a PART SE to one or more design variables from other PART SEs.



Format:

1	2	3	4	5	6	7	8	9	10
SEDLINK	ID	DSEID	DDVID	C0	CMULT	ISEID1	IDV1	C1	
	ISEID2	IDV2	C2	ISEID3	IDV3	C3			
	ISEID4	IDV4	C4	-etc.-					

Field	Contents
ID	Unique identification number. (Integer > 0)
DSEID	PART SE identification number for DDVID (Integer ≥ 0)
DDVID	Dependent design variable identification number. (Integer > 0)
C0	Constant term. (Real; Default = 0.0)
CMULT	Constant multiplier. (Real; Default = 1.0)
ISEIDi	PART SE identification number for IDVi (Integer ≥ 0)
IDVi	Independent design variable identification number. (Integer > 0)
Ci	Coefficient i corresponding to IDVi. (Real)

The SEDLINK bulk data entry enables linking design variables across superelement boundaries. It resembles the DLINK entry and the discussion for that entry applies here as well. The distinction is that it is now necessary to specify the parts superelement of the independent design variable(s) as well as its ID. The SEDLINK must appear in the main bulk data section and is ignored if it is placed after a BEGIN SUPER=seid data packet, where seid>0.

SEDRSP2

Purpose:

Defines equation responses that are used in the design, either as constraints or as an objective with quantities from multiple PART SEs.



Format:

	1	2	3	4	5	6	7	8	9	10
SEDRSP2	ID	LABEL	EQID or FUNC	REGION	METHOD	C1	C2	C3		
"DESVAR"	DVSEID1	DVID1	DVSEID2	DVID2	DVSEID3	DVID3				
	DVSEID4	DVID4	-etc.-							
"DTABLE"	LBSEID1	LABL1	LBSEID2	LABL2	LBSEID3	LABL3				
	LBSEID4	LABL4	-etc.-							
"DRESP1"	R1SEID1	NR1	R1SEID2	NR2	R1SEID3	NR3				
	R1SEID4	NR4	-etc.-							
"DNODE"	NDSEID1	G1	CMP1	NDSEID2	G2	CMP2				
	NDSEID3	G3	CMP3	-etc.-						
"DVPREL1"	P1SEID1	DPIP1	P1SEID2	DPIP2	P1SEID3	DPIP3				
	P1SEID4	DPIP4	-etc.-							
"DVCREL1"	C1SEID1	DCIC1	C1SEID2	DCIC2	C1SEID3	DCIC3				
	C1SEID4	DCIC4	-etc.-							
"DVMREL1"	M1SEID1	DMIM1	M1SEID2	DMIM2	M1SEID3	DMIM3				
	M1SEID4	DMIM4	-etc.-							
"DVPREL2"	P2SEID1	PDI2P1	P2SEID2	DPI2P2	P2SEID3	DPI2P3				
	P2SEID4	DPI2P4	-etc.-							
"DVCREL2"	C2SEID1	DC12C1	C2SEID2	DC12C2	C2SEID3	DC12C3				
	C2SEID4	DC12C4	-etc.-							
"DVMREL2"	M2SEID1	DMI2M1	M2SEID2	DMI2M2	M2SEID3	DMI2M3				
	M2SEID4	DMI2M4	-etc.-							

Field	Contents
ID	Unique identification number. (Integer > 0)
LABEL	User-defined label. (Character)
EQID	DEQATN entry identification number. (Integer > 0)
FUNC	Function to be applied to the arguments. (Character)
REGION	Region identifier for constraint screening. (Integer > 0)
METHOD	When used with FUNC = BETA, METHOD = MIN indicates a minimization task while MAX indicates a maximization task. (Default = MIN) When used with FUNCT = MATCH, METHOD = LS indicated a least squares while METHOD = BETA indicated minimization of the maximum difference. (Default = LS)
Ci	Constants used when FUNC = BETA or FUNC = MATCH in combination with METHOD = BETA. See Remark 8. (Real; Defaults: C1 = 1.0., C2 = .005, and C3=10.0)



Field	Contents
“DESVAR”	Flag indicating DESVAR entry identification numbers. (Character)
DVSEIDi	PART SE identification number for DVIDi (Integer ≥ 0)
DVIDi	DESVAR entry identification number. (Integer > 0)
“DTABLE”	DTABLE flag. Indicates that the LABLs for the constants in a DTABLE or DTABLE2 entry follow. This field may be omitted if there are no constants involved in this relation. (Character)
LBSEIDj	PART SE identification number for LABLj. (Integer ≥ 0)
LABLi	Label for a constant on the DTABLE or DTABLE2 entry. (Character)
“DRESP1”	Flag indicating DRESP1 entry identification numbers. (Character)
R1SEIDk	PART SE identification number for NRk. (Integer ≥ 0)
NRk	DRESP1 entry identification number. (Integer > 0)
“DNODE”	Flag indicating grid point and component identification numbers. (Character)
NDSEIDm	PART SE identification number for (Gm,Cm). (Integer ≥ 0)
Gm	Identification number for any grid point in the model. (Integer > 0)
Cm	Component number of grid point Gm. (1 < Integer < 3)
“DVPREL1”	Flag indicating DVPREL1 entry identification number. (Character)
P1SEIDi	PART SE identification number for DPPIPi. (Integer ≥ 0)
DPPIPi	DVPREL1 entry identification number. (Integer > 0)
“DVCREL1”	Flag indicating DVCREL1 entry identification number. (Character)
C1SEIDi	PART SE identification number for DCICi. (Integer ≥ 0)
DCICi	DVCREL1 entry identification number. (Integer > 0)
“DVMREL1”	Flag indicating DVPREL2 entry identification number. (Character)
M1SEIDi	PART SE identification number for DMIMi. (Integer ≥ 0)
DMIMi	DVMREL1 entry identification number. (Integer > 0)
“DVPREL2”	Flag indicating DVPREL2 entry identification number. (Character)
P2SEIDi	PART SE identification number for DPI2Pi. (Integer ≥ 0)
DPI2Pi	DVPREL2 entry identification number. (Integer > 0)
“DVCREL2”	Flag indicating DVCREL2 entry identification number. (Character)
C2SEIDi	PART SE identification number for DCI2Ci. (Integer ≥ 0)
DCI2Ci	DVCREL2 entry identification number. (Integer > 0)
“DVMREL2”	Flag indicating DVMREL2 entry identification number. (Character)
M2SEIDi	PART SE identification number for DMI2Mi. (Integer ≥ 0)
DMI2Mi	DVMREL2 entry identification number. (Integer > 0)



The SEDRSP2 entry closely resembles the DRESP2 entry in that it combines design quantities and user defined constants to create a new response that is not otherwise available using a DEQATN or an embedded function. The comments for the DRESP2 therefore apply to the SEDRSP2, but now the combined quantities can be gathered from separate parts superelements. The SEDRSP2 must appear in the main bulk data section and is ignored if it is placed after a BEGIN SUPER=seid data packet, where seid>0. Unlike the DRESP2, the SEDRSP2 does not support DRESP2 or DVREL1 flags. It is recommended that DTABLE constants be placed in the main bulk data section. If they are in the parts superelement and they are referenced by the SEDRSP2, there must be at least one DESVAR, DVxRELY or DRESP1 Flag in the SEDRSP2 that references the same SEID.

SEDRSP3

Purpose:

Defines constituents from multiple PART SE for an external response using user-supplied routine(s).



Format:

1	2	3	4	5	6	7	8	9	10
SEDRSP3	ID	LABEL	GROUP	TYPE	REGION				
	"DESVAR"	DVSEID1	DVID1	DVSEID2	DVID2	DVSEID3	DVID3		
		DVSEID4	DVID4	-etc.-					
	"DTABLE"	LBSEID1	LABL1	LBSEID2	LABL2	LBSEID3	LABL3		
		LBSEID4	LABL4	-etc.-					
	"DRESP1"	R1SEID1	NR1	R1SEID2	NR2	R1SEID3	NR3		
		R1SEID4	NR4	-etc.-					
	"DNODE"	NDSEID1	G1	CMP1	NDSEID2	G2	CMP2		
		NDSEID3	G3	CMP3	-etc.-				
	"DVPREL1"	P1SEID1	DPIP1	P1SEID2	DPIP2	P1SEID3	DPIP3		
		P1SEID4	DPIP4	-etc.-					
	"DVCREL1"	C1SEID1	DCIC1	C1SEID2	DCIC2	C1SEID3	DCIC3		
		C1SEID4	DCIC4	-etc.-					
	"DVMREL1:	M1SEID1	DMIM1	M1SEID2	DMIM2	M1SEID3	DMIM3		
		M1SEID4	DMIM4	-etc.-					
	"DVPREL1"	P2SEID	DPI2P1	P2SEID2	DPI2P2	P2SEID3	DPI2P3		
		P2SEID4	DPI2P4	-etc.-					
	"DVCREL2"	C2SEID1	DCI2C1	C2SEID2	DCI2C2	C2SEID3	DCI2C3		
		C2SEID4	DCI2C4	-etc.-					
	"DVMREL2"	M2SEID	DMI2M1	M2SEID2	DMI2M2	M2SEID3	DMI2M3		
		M2SEID4	DMI2M4	-etc.-					
	"USRDATA"				String				
					-etc.-				

Field	Contents
ID	Unique identification number. (Integer > 0)
LABEL	User-defined label. (Character)
GROUP	Group name the external response type belongs to (Character).
TYPE	External response type (Character).
“DESVAR”	Flag indicating DESVAR entry identification numbers. (Character)
DVSEIDI	PART SE identification number for DVIDi. (Integer ≥ 0)
DVIDi	DESVAR entry identification number. (Integer > 0)
“DTABLE”	DTABLE flag. Indicates that the LABLs for the constants in a DTABLE or DTABLE2 entry follow. This field may be omitted if there are no constants involved in this relation. (Character)
LBSEIDj	PART SE identification number for LABLj. (Integer ≥ 0)



Field	Contents
LABLi	Label for a constant on the DTABLE or DTABLE2 entry. (Character)
“DRESP1”	Flag indicating DRESP1 entry identification numbers. (Character)
R1SEIDk	PART SE identification number for NRk. (Integer ≥ 0)
NRk	DRESP1 entry identification number. (Integer > 0)
“DNODE”	Flag indicating grid point and component identification numbers. (Character)
NDSEIDm	PART SE identification number for (Gm,Cm). (Integer ≥ 0)
Gm	Identification number for any grid point in the model. (Integer > 0)
Cm	Component number of grid point Gm. ($1 < \text{Integer} < 3$)
“DVPREL1”	Flag indicating DVPREL1 entry identification number. (Character)
P1SEIDI	PART SE identification number for DPiPi. (Integer ≥ 0)
DPIPi	DVPREL1 entry identification number. (Integer > 0)
“DVCREL1”	Flag indicating DVCREL1 entry identification number. (Character)
C1SEIDI	PART SE identification number for DCICi. (Integer ≥ 0)
DCICi	DVCREL1 entry identification number. (Integer > 0)
“DVMREL1”	Flag indicating DVMREL2 entry identification number. (Character)
M1SEIDI	PART SE identification number for DMIMi. (Integer ≥ 0)
DMIMi	DVMREL1 entry identification number. (Integer > 0)
“DVPREL2”	Flag indicating DVPREL2 entry identification number. (Character)
P2SEIDI	PART SE identification number for DPI2Pi. (Integer ≥ 0)
DPI2Pi	DVPREL2 entry identification number. (Integer > 0)
“DVCREL2”	Flag indicating DVCREL2 entry identification number. (Character)
C2SEIDI	PART SE identification number for DCI2Ci. (Integer ≥ 0)
DCI2Ci	DVCREL2 entry identification number. (Integer > 0)
“DVMREL2”	Flag indicating DVMREL2 entry identification number. (Character)
M2SEIDI	PART SE identification number for DMI2Mi. (Integer ≥ 0)
DMI2Mi	DVMREL2 entry identification number. (Integer > 0)
“USRDATA”	Flag indicating user input data. (Character).

The SEDRSP3 entry closely resembles the DRESP3 entry in that it combines design quantities and user defined constants to create a new response that is not otherwise available using a user supplied external function. The comments for the DRESP3 therefore apply to the SEDRSP3, but now the combined quantities can be gathered from separate parts superelements. The SEDRSP3 must appear in the main bulk data section and is ignored if it is placed after a BEGIN SUPER=seid data packet, where seid>0. Unlike the DRESP3, the SEDRSP3 does not support DRESP2 or DVLREL1 flags. It is recommended that DTABLE constants be placed in the main bulk data section. If they are in the parts superelement and



they are referenced by the SEDRSP3, there must be at least one DESVAR, DVxRELY or DRESP1 Flag in the SEDRSP that references the same SEID.

STOCHAS

Purpose:

Specifies statistics used in randomization selected model parameters.

Format:

1	2	3	4	5	6	7	8	9	10
STOCHAS	SID	PENTRY	CDF	CoV	m				
		MENTRY	CDF	CoV	m				
		CENTRY	CDF	CoV	m				
		LOADS	CDF	CoV	m				
		SPCD	CDF	CoV	m				

Example 1:

Randomize all element and material properties with the default settings

STOCHAS	100	PENTRY							
		MENTRY							

Example 2:

Randomize material properties with CoV = 0.1 and loadings with CoV = 0.3 and default multipliers of standard deviations

STOCHAS	200	LOADS	GAUSS	0.3					
		MENTRY	GAUSS	0.1					

Field	Contents
SID	Unique identification number that is selected by the STOCHASTICS Case Control command. (Integer > 0)
"PENTRY"	Flag for randomizing real values on all the element property entries. (Character)
"MENTRY"	Flag for randomizing real values on all the independent material property entries. (Character)
"CENTRY"	Flag for randomizing real values on all the connectivity entries. (Character)
"LOADS"	Flag for randomizing real values on all the load entries. (Character)
"SPCD"	Flag for randomizing real values on all the SPCD entries. (Character)



Field	Contents
CDF	Name of a cumulative distribution function. See Remark 2. (Character; Default = GAUSSIANS or blank).
CoV	Coefficient of variance. (Real > 0; Default = 0.05)
m	Number of standard deviations. See Remark 3. (Real > 0; Default = 3.0)

Discussion:

The STOCHAS entry was provided to simulate the condition that user defined parameters are not known exactly, but have a statistical variation. The intent is to assess the variability in the results due to the variability in the input. This is a “stand-alone” computation and it does not interact with standard SOL 200 capabilities of sensitivity or optimization. Further, it is available in all the linear solution sequences of MSC Nastran and is listed here only because it bears some similarity to the design of properties provided by the DVxRELY entries. Some guidelines in applying the feature are:

1. At least one flag must exist but they can be placed in any order.
2. Only Gaussian distributions are supported.
3. The range of a random variable is defined as $(\mu - m \cdot \sigma, \mu + m \cdot \sigma)$ where μ is the mean of the random variable (or the value of one analysis model parameter on a Bulk Data entry), σ is the standard deviation that is related to μ , CoV by $\sigma = CoV \cdot \mu$ and m is the multiplier of the standard deviations.
4. $m \cdot CoV$ must be < 1.0 .

TOMVAR**Purpose:**

Defines a design region for topometry optimization (element-by-element optimization).

Entry Description:

1	2	3	4	5	6	7	8	9	10
TOMVAR	ID	TYPE	PID	PNAME/ FID	XINIT	XLB	XUB	DELXV	
	DLINK	TID	C0	C1					
	DDVAL	DSVID							

Field	Contents
ID	Unique topometry design region identification number. (Integer > 0)
TYPE	Property entry type. Used with PID to identify the elements to be designed. (Character: “PBAR”, “PSHELL”, ‘PSOLID’, and “PCOMP”, etc.)



Field	Contents
PID	Property entry identifier (Integer > 0). This PID must be unique for PIDs referenced by other TOPVAR, DVPREL1, DVPREL2, DVMREL1, and DVMREL2 entries. Topometry, topology, and sizing variables cannot share the same properties. (Integer > 0). Combined topometry, topology, topography, sizing, and shape variables are allowed.
PNAME/FID	Property name or property material name, such as "T", "A", "E", and "GE", or field position of the property entry or word position in the element property table of the analysis model. Property names that begin with an integer such as 12I/T**3 may only be referenced by field position. (Character or Integer > 0.)
XINIT	Initial value. (Real or blank, no Default). Typically, XINIT is defined to match the mass target constraint (so the initial design does not have violated constraints) or the analysis model input property value.
XLB	Lower bound. (Real or blank; Default = blank) . The default is XLB=0.5*XINIT.
XUB	Upper bound . (Real or blank; Default = blank). The default is XLB=1.5*XINIT.
DELXV	Fractional change allowed for the design variable during approximate optimization. (Real > 0.0; Default = 0.5)
DDVAL	Indicates that this line defines discrete TOMVAR variables.
DSVID	DDVAL entry identifier. (Integer > 0)
DLINK	Indicates that this line relates a ply thickness to another ply thickness.
TID	TOMVAR entry identifier. (Integer > 0)
C0	Constant term. (Real; Default = 0.0)
C1	Coefficient term. (Real; no Default)

The TOMVAR entry provides a simple user interface that allows you to easily generate a design model that independently designs each element referenced by the TOMVAR. For example if TYPE is PSHELL, PID is 1000 and PNAME is T, the thickness of each element that invokes PSHELL 1000 will be independently designed. A special feature of the entry is that the PNAME can refer to a material property that is associated with the PID. So if TYPE is PSHELL, PID is 1000 and PNAME is E, Young's modulus will be separately designed for each element that is associated with PSHELL 1000. This feature creates a problem in one instance: PNAME="A" could mean the area of a property or it could be the thermal expansion coefficient on a MATI entry. This is resolved by always interpreting PNAME="A" as a thermal expansion coefficient. If the area of a property is desired, the integer field ID for that property is used. The TOMVAR supports DESVAR like features, such as discrete variables and design variable linking. The linking is limited to ply thicknesses on the PCOMP or PCOMPG entry.



TOPVAR

Purpose:

Defines a topology design region for topology optimization.

Entry Description:

1	2	3	4	5	6	7	8	9	10
TOPVAR	ID	LABEL	PTYPE	XINIT	XLB	DELXV	POWER	PID	
	“SYM”	CID	MS1	MS2	MS3	CS	NCS		
	“CAST”	CID	DD	DIE	ALIGN				
	“EXT”	CID	ED	ALIGN					
	“TDMIN”	TVMIN	TVMAX						
	“STRESS”	STLIM							

Field	Contents
ID	Unique topology design region identification number. (Integer > 0)
LABEL	User-supplied name for printing purpose. (Character)
PTYPE	Property entry name. Used with PID to identify the elements to be designed. (Character: "PBAR", "PSHELL", "PSOLID", etc.)
XINIT	Initial value. (Blank or Real, XLB < XINIT ≤ 1.0; Default = Blank). Typically, XINIT is defined to match the mass constraint on DRESP1=FRMASS, so the initial design does not have violated constraints. In this case, the default is set to the constraint value. If the mass (DRESP1=FRMASS or WEIGHT) is the objective, the default is 0.9. The default of XINIT is 0.6 for the other cases.
XLB	Lower bound to prevent the singularity of the stiffness matrix. (Real; Default = 0.001)
DELXV	Fractional change allowed for the design variable during approximate optimization. See Remark 3. (Real > 0.0; Default = 0.2)
POWER	A penalty factor used in the relation between topology design variables and element Young's modulus. (Real > 1.0; Default = 3.0). 2.0 < POWER < 5.0 is recommended.
PID	Property entry identifier. This PID must be unique for PIDs referenced by other TOPVAR, DVPREL1 and DVPREL2 entries. Topology and sizing variables cannot share the same properties. (Integer > 0)
“SYM”	Indicates that this line defines symmetry constraints.
CID	Rectangular coordinate system ID used for specifying manufacturing constraints. See Remark 4. (Blank or Integer > 0; Default = blank)
MSi	Mirror symmetry plane. (Character, 'XY', 'YZ', or 'ZX')
CS	Cyclic symmetry axis. (character X, Y, Z).
NCS	Number of cyclic symmetric segments in 360 degrees (Integer > 0).



Field	Contents
"CAST"	Indicates that this line defines casting constraints (i.e., die draw direction constraints).
DD	Draw Direction. DDi=X, Y, Z or X-, Y-, Z- for a single die option (DIE=1) where X-, Y-, Z- indicates the opposite direction of X, Y, and Z respectively. DDi=X, Y, and Z for two die option (DIE =2) (Character)
DIE	Die Options. (Blank or integer 1 or 2; Default = 1) = 1 (or blank). A single die will be used and the die slides in the given draw direction (i.e., material grows from the bottom in the draw direction) = 2. Two dies will be used and the dies split apart along the draw direction (i.e., material grows from the splitting plane in opposite direction along the axis specified by the draw direction DDi. The splitting plane is determined by optimization)
ALIGN	Indicates whether the designed property finite element mesh is precisely aligned with the draw direction or extrusion direction. (Character: "YES" or "NO" or Blank; Default = blank = "NO")
"EXT"	Indicates that this line defines extrusion constraints (i.e., enforce constant cross-section)
ED	Extrusion direction. (Character, X, Y, or Z)
"TDMIN"	Indicates that this line defines a minimum and/or maximum member size.
TVMIN	Minimum member size. (Real \geq 0.0 or blank)
TVMAX	Maximum member size. (Real $>$ TVMIN or blank)
"STRESS"	Indicates that this line defines a stress limit.
STLIM	von Mises stress upper bound. (Real >0.0)

Discussion:

Topology optimization is a special version of optimization that seeks to find the optimal distribution of material based on some criterion for given space, loads, and boundary conditions. The TOPVAR entry identifies the finite elements that are to participate in the design task. Every element that is a member of the PTYPE/PID pair given of the TOPVAR has its own design variable that drives the Young's modulus and density of the element. The desired case is that the design variable will be driven to either its upper bound value of 1.0 (retain the element) or its lower bound values close to zero (discard the element). In addition to specifying the PTYPE/PID information, the entry provides an initial value for the variable upper and lower bound, a DELXV parameter to specify how rapidly the design variable can change, and a POWER parameter that is the exponent used in [Equation \(7-1\)](#) (as given in [Topology Optimization](#)). Typically, the default values for these parameters are adequate.

The PTYPES than can be invoked by the TOPVAR include PROD, PBAR, PBARL, PBEND, PBEAM, PBEAML, PSHELL, PSHEAR, PSOLID, and PWELD. All designated element properties must refer to a MAT1 entry; therefore, a PCOMP cannot be designed in topology optimization.

The optional continuation lines for this entry permit the specification of manufacturability constraints as defined as given in the [Topology Optimization](#)). Manufacturability constraints are specified in a rectangular (CORD1R and CORD2R) coordinate system and only one system can be specified for a given TOPVAR entry. One, two or three different mirror symmetry planes can be present (such as



MS1=XY, MS2=YZ, and MS3=ZX). Casting (“CAST”) and Extrusion (“EXT”) manufacturability constraints can be applied to PTYPE=”PSOLID” only. Casting constraints cannot be combined with extrusion constraints for the same TOPVAR entry.

Some symmetry constraint types can be combined with casting or extrusion constraints. The referenced coordinate system CID must be the same for the combined constraints. Some possible combinations are:

- For “EXT” constraints, possible combinations are (Edi=X, MSi=XY and/or ZX), (Edi=Y, MSi=YZ and/or XY), (Edi=Z, MSi=ZX and/or YZ),
- For “CAST” constraints, possible combinations are (DDi=X or -X, MSi=XY and/or ZX), (DDi=Y or -Y, MSi=YZ and/or XY), (DDi=Z or -Z, MSi=ZX and/or YZ)

For the two dies option (DIE=2), the splitting plane is optimized. For a single die DIE=1, the parting plane is the bottom surface of the designed part in the draw direction.

TDMIN is a dimensional quantity with a guideline that it should be set to at least three times a representative element dimension. Without a TDMIN continuation line, the minimum member size constraint is taken from the specification of TDMIN parameter on the DOPTPRM entry. This option is applied on 2 and 3 D elements only. Minimum member size constraints can be used with “SYM”, “CAST”, and “EXT” constraints. TVMAX is a dimensional quantity that should be no greater than ten times a representative element dimension. This parameter is reserved for special situations where the designs would otherwise pose manufacturing difficulties due to large member size.

For problems with casting constraints, it is recommended to use an aligned mesh for the topology designed property since a smaller tolerance can be used.

STLIM imposes stress limits on the designed and undersigned solid elements. Stress limits are not supported for one and two dimensional elements.

[Topology Optimization](#) contains a number of simple Topology Optimization examples that demonstrate various manufacturability constraints.



Parameters Unique to Design Sensitivity and Optimization

MSC Nastran allows you to specify a number of parameters for the input of scalar values and for requesting special features. These are in addition to the optimization control possible with Bulk Data entries such as DOPTPRM and DSCREEN. This section is intended to be a complete quick reference for all such Case Control and Bulk Data parameters that are associated with Design Sensitivity and Optimization. Refer to [Parameters](#) (p. 725) in the *MSC Nastran Quick Reference Guide* for a description of all MSC Nastran parameters.

AUTOADJ	Select the adjoint sensitivity analysis.
YES	Selection is based on a determination as whether the adjoint or the direct method is most efficient. (Default)
NO	Direct method is used.
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.
DESPCH1	Specifies the amount of data to be written to the .pch file. (Default = 6)
N = 0	Write no data.
N = \pm 1	Write property entries that are designed.
N = \pm 2	Write all property entries of a given type when one or more property of that type is designed.
N = \pm 4	Write new DESVAR and DRESP1 entries.
N = \pm n	Write combined quantities from summing values.
Note:	A positive value of N results in large field formats while a negative value gives small field formats.
DPEPS	Controls overriding of the analysis value by the design model. (Default = 1.0E-4)
	If all the differences between the analysis and design model properties are less than DPEPS, the differences are ignored and it is possible to reuse analysis results from a previous run. Setting this parameter to a large value is not recommended since it can result in poor sensitivity values.



DSNOKD	Indicates if the differential stiffness effect is to be included in buckling sensitivity analysis.
1.0	Include differential stiffness.
0.0	Do not include differential stiffness. (Default)
DSZERO	Specifies the minimum absolute value for printout of design sensitivities. (Default = 0.0)
IUNIT	IUNIT specifies the FORTRAN unit number on which the DSCMCOL table and the DSCM2 matrix in SOL 200 will be written. (See the OUTPUT2 and OUTPUT4 module descriptions in the <i>MSC Nastran 2005 r3 DMAP Programmer's Guide</i> .) (Default = 11)
NASPART	Controls the frequency of MSC 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 seven predetermined exit points.
0	Do not exit. Proceed with optimization. (Default)
$N > 0$	Exit at one of the locations 1 through 7 ($1 \leq N \leq 7$).
1	Exit after the initialization of the analysis and design model but before finite element analysis begins.
2	Exit after finite element analysis and initial design response and shape basis vector processing.
3	Exit after design constraint evaluation and screening.
4	Exit after design sensitivity analysis and print the matrix of design sensitivity coefficients (DSCM2). This is equivalent to the DSAPRT (UNFORM,END=SENS) Case Control command.
-4	Exit after design sensitivity analysis and write the data blocks related to sensitivity coefficients (DSCM2 and DSCMCOL) to an external file using the OUTPUT2 and OUTPUT4 modules. This is equivalent to the DSAPRT (NOPRINT,EXPORT END=SENS) Case Control command. See related parameters ITAPE, IUNIT, and OMAXR.
5	Exit after the first approximate optimization of the design model.
6	Exit after the first update of the analysis model based on the first approximate optimization of the design model.



	7	Compute and output design sensitivity coefficients at the end of normal program termination: hard convergence, soft convergence, or maximum design cycles. This is equivalent to the DSAPRT (UNFORM,START=LAST) Case Control command. If the final design is a discrete design optimization, no sensitivity is performed and the OPTEXIT=7 request is not honored.
SENSUOO		If set to yes, pseudo-displacements are recovered to the o-set while including the terms of Equation (2-69) . Default=NO
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.
SOLADJC		Indicates if adjoint solution vectors are to be calculated during the analysis. -1 Do not calculate adjoint solution vectors during the analysis. ≥ 0 For ANALYSIS = DFREQ, the adjoint vectors will be calculated during the solution if the conditions given in the full description in Parameters (p. 725) in the <i>MSC Nastran Quick Reference Guide</i> are met. (Default)
TDMIN	Default = 0.0	Topology minimum member diameter in the basic coordinate system. (Real > 0.0 , Default = 0.0, i.e., no minimum member size control.) This option is applied on 2D and 3D elements only.
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 grids). UPDTBSH only controls updates on the boundary shapes.
XYUNIT	n	Specifies the unit number for storage of design sensitivity data in comma-separated value format.



Design Responses

DRESP1

This section contains a detailed description of each of the Response Types in the DRESP1 entry.

In order to facilitate this discussion, [Table 4-5](#) is duplicated from the Bulk Data entry **DRESP1** (p. 1828) in the *MSC Nastran Quick Reference Guide*. The “Remarks” that are called out in this table are not given explicitly here, but the information is provided in detail in the response type descriptions that follow.

Table 4-5 Design Sensitivity Response Attributes

Response Type (RTYPE)	Response Attributes		
	ATTA (Integer > 0)	ATTB (Integer > 0 or Real > 0.0)	ATTI (Integer > 0)
WEIGHT	Row Number ($1 \leq \text{ROW} \leq 6$) See Remark 24 .	Column Number ($1 \leq \text{COL} \leq 6$)	SEIDi or All or blank. See Remark 12 .
VOLUME	Blank	Blank	SEIDi or ALL or blank. See Remark 12 .
FRMASS (see Remarks 28 . & 29 .)	Blank	Blank	Blank or Property ID (PID). See Remark 37 .
COMP (see Remark 28 .)	Blank	Blank	Blank
EIGN	Normal Modes Mode Number. See Remark 33 .	Approximation Code. See Remark 19 .	Blank
CEIG	Complex Eigenvalue Mode Number. (Integer > 0)	ALPHA or OMEGA (Default = ALPHA)	Blank
FREQ	Normal Modes Mode Number. See Remarks 18 . and 33 .	Approximation Code. See Remark 19 .	Blank
LAMA	Buckling Mode Number	Approximation Code. See Remark 19 .	Blank
DISP	Displacement Component	Blank or Mode Number	Grid ID
STRAIN	Strain Item Code	Blank or Mode Number	Property ID (PID) or Element ID (EID)



Table 4-5 Design Sensitivity Response Attributes

Response Type (RTYPE)	Response Attributes		
	ATTA (Integer > 0)	ATTB (Integer > 0 or Real > 0.0)	ATTI (Integer > 0)
ESE	Strain Energy Item Code See Remark 21.	Blank or Mode Number	Property ID (PID) or Element ID (EID)
STRESS	Stress Item Code	Blank or Mode Number	Property ID (PID) or Element ID (EID)
FORCE	Force Item Code	Blank or Mode Number	Property ID (PID) or Element ID (EID)
FATIGUE See Remark 39.	Fatigue Item Code. See Remark 43.	ID of a FATIGUE case control. See Remark 40.	Property ID (PID) or Element ID (EID)
SPCFORCE	SPC Force Component	Blank	Grid ID
CSTRAIN	Strain Item Code	LAMINA Number (Integer; Default = 1)	Property ID (PID) or Element ID (EID)
CSTRESS	Stress Item Code	LAMINA Number (Integer; Default = 1)	Property ID (PID) or Element ID (EID)
CFAILURE	Failure Indices Item Code	LAMINA Number (Integer; Default = 1)	Property ID (PID) or Element ID (EID)
CSTRAT	Composite Strength Ratio Item Code	LAMINA Number (Integer; Default = 1)	Property ID (PID) or Element ID (EID)
TOTSE (Total Strain Energy)	Blank	Blank or Mode Number	SEIDI or All or blank. See Remark 12.
GPFORCE	GPFORCE Component Code(1-6; see Remark 25.)	Blank	Element ID
GPFORCP	Grid Point (see Remark 26.)	Blank	Grid ID connected to ATTA grid to specify orientation.
ABSTRESS	Arbitrary Beam Stress Item Code (see Remark 30.)	Blank	Property ID (PID) or Element ID (EID)
FRDISP	Displacement Component	Frequency Value. (Blank; Real \geq 0.0 or Character) See Remarks 15. and 20.	Grid ID



Table 4-5 Design Sensitivity Response Attributes

Response Type (RTYPE)	Response Attributes		
	ATTA (Integer > 0)	ATTB (Integer > 0 or Real > 0.0)	ATTI (Integer > 0)
PRES	Acoustic Pressure Component (= 1 or 7)	Frequency Value. (Blank; Real ≥ 0.0 or Character) See Remarks 15. and 20.	Grid ID
FRVELO	Velocity Component	Frequency Value. (Blank; Real ≥ 0.0 or Character) See Remarks 15. and 20.	Grid ID
FRACCL	Acceleration Component	Frequency Value. (Blank; Real ≥ 0.0 or Character) See Remarks 15. and 20.	Grid ID
FRSPCF	SPC Force Component	Frequency Value. (Blank; Real ≥ 0.0 or Character) See Remarks 15. and 20.	Grid ID
FRSTRE	Stress Item Code	Frequency Value. (Blank; Real ≥ 0.0 or Character) See Remarks 15. and 20.	Property ID (PID) or Element ID (EID)
FRFORC	Force Item Code	Frequency Value. (Blank; Real ≥ 0.0 or Character) See Remarks 15. and 20.	Property ID (PID) or Element ID (EID)
PSDDISP	Displacement Component (see Remarks 27. and 31.)	Frequency Value. (Blank; Real ≥ 0.0 or Character). See Remarks 15. and 20.	Grid ID
PSDVELO	Velocity Component (see Remarks 27. and 31.)	Frequency Value (Blank; Real ≥ 0.0 or Character). See Remarks 15. and 20.	Grid ID
PSDACCL	Acceleration Component (see Remarks 27. and 31.)	Frequency Value. (Blank; Real ≥ 0.0 or Character). See Remarks 15. and 20.	Grid ID



Table 4-5 Design Sensitivity Response Attributes

Response Type (RTYPE)	Response Attributes		
	ATTA (Integer > 0)	ATTB (Integer > 0 or Real > 0.0)	ATTI (Integer > 0)
RMSDISP	Displacement Component (see Remark 31.)	RANDPS ID	Grid ID
RMSVELO	Velocity Component (see Remark 31.)	RANDPS ID	Grid ID
RMSACCL	Acceleration Component (see Remark 31.)	RANDPS ID	Grid ID
ACPWR - acoustic power radiated through a panel. (See Remark 34.)	Blank	Frequency value. (Blank for all forcing frequency; Real ≥ 0.0)	Blank
ACINTS - acoustic intensity	Blank	Frequency value. (Blank for all forcing frequency; Real ≥ 0.0)	Grid ID of wetted surface.
AFPRES - Acoustic Intensity for AFPM. (See Remark 35.)	Acoustic Pressure Component (Integer = 1 or 7)	Frequency value (Blank for all forcing frequency; Real ≥ 0.0)	Grid ID of AFPMID.
AFINTS - Acoustic Intensity for AFPM. (See Remark 35.)	Component Code - 0 - normal to AFPM, 1 - x-dir 2 - y-dir 3 - z-dir	Frequency value. (Blank for all forcing frequency; Real ≥ 0.0)	Grid ID of AFPMID.
AFVELO - Velocity for AFPM. (See Remark 35.)	Component Code - 11 - Real/Mag in x-dir 12 - Real/Mag in y-dir 13 - Real/Mag in z-dir 71 - Img/Ph in x-dir 72 - Img/Ph in y-dir 73 - Img/Ph in z-dir	Frequency value. (Blank for all forcing frequency; Real ≥ 0.0)	Grid ID of AFPMID.
AFPWR - Acoustic Power for AFPM. (See Remark 35.)	Blank	Frequency value. (Blank for all forcing frequency; Real ≥ 0.0)	Blank
ERP (See Remarks 41. and 42.)	ERP Item Code	Frequency Value. (Blank; Real ≥ 0.0 or Character) See Remarks 15. and 20.	Set3 ID or Blank



Table 4-5 Design Sensitivity Response Attributes

Response Type (RTYPE)	Response Attributes		
	ATTA (Integer > 0)	ATTB (Integer > 0 or Real > 0.0)	ATTI (Integer > 0)
TDISP	Displacement Component	Time Value. (Blank; Real; or Character) See Remarks 16. and 20.	Grid ID
TVELO	Velocity Component	Time Value. (Blank; Real, or Character) See Remarks 16. and 20.	Grid ID
TACCL	Acceleration Component	Time Value. (Blank; Real, or Character) See Remarks 16. and 20.	Grid ID
TSPCF	SPC Force Component	Time Value. (Blank; Real, or Character) See Remarks 16. and 20.	Grid ID
TSTRE	Stress Item Code	Time Value. (Blank; Real, or Character) See Remarks 16. and 20.	Property ID (PID) or Element ID (EID)
TFORC	Force Item Code	Time Value. (Blank; Real, or Character) See Remarks 16. and 20.	Property ID (PID) or Element ID (EID)
STMONP1 Structural MONPNT1	Component (see Remark 36.)	Blank	Blank
STMOND1 Structural MONDSP1	Component (see Remark 36.)	Blank	Blank
MONPNT3	Component (see Remark 36.)	Blank	Blank
AEMONP1 Aerodynamic MONPNT1	Component (see Remark 36.)	Blank	Blank
AEMOND1 Aerodynamic MONDSP1	Component (see Remark 36.)	Blank	Blank
TRIM	AESTAT or AESURF entry ID	Blank	Blank
STABDER	AESTAT or AESURF entry ID	Restraint Flag. (Integer 0 or 1) See Remark 13.	Component



Table 4-5 Design Sensitivity Response Attributes

Response Type (RTYPE)	Response Attributes		
	ATTA (Integer > 0)	ATTB (Integer > 0 or Real > 0.0)	ATTI (Integer > 0)
FLUTTER	Blank	Blank	See Remark 14.
DIVERG	Divergence Root Number (See remark 38.)	Blank	Mach No.
WMPID (see Remarks 44. and 45.)	MID	SEID	PID

WEIGHT

The weight response is computed using the same Grid Point Weight Generator utility that is used when with the PARAM GRDPNT request is used to obtain printed results regarding the weight. This calculation results in a 6x6 mass matrix computed about the grid designated by the GRDPNT or about the origin of the basic coordinate system if the GRDPNT is not defined (or is defined and the grid point does not exist). The particular weight that is to be used in the response is selected with the ATTA field specifying which row of the 6x6 matrix field to select and ATTB field identifying the column. The default values for ATTA and ATTB are 3, resulting in the mass of the structure in the z direction being the default weight response. The associated ANALYSIS Type column of [Table 4-3](#) indicates that this response can be invoked from any ANALYSIS type or above the subcase level before an ANALYSIS type is specified. It is recommended that the response be invoked above the subcase level using the DESOBJ Case Control command if it is the response is the objective and the DESGLB Case Control command if the response is constrained. The response can be for the entire structure or can be for an individual superelement specified in the ATT1 field. If ATT1 is blank, the total weight of the structure is used for the response. If it is zero, the weight of the residual is used.

The following bulk data fragment shows how three DRESP1's with response type WEIGHT can be combined to compute the x and y locations of the center of gravity of a structure. This enables using the center of gravity value as a design constraint or the objective.

```

$ 
DRESP1      33      WT33      WEIGHT          3      3
DRESP1      26      W26      WEIGHT          2      6
DRESP1      34      W34      WEIGHT          3      4
$ 
$ THE CENTER OF GRAVITY (CG)
$ 
DRESP2      50      CG-X     200
DRESP1      41      43
DRESP2      51      CG-Y     200
DRESP1      41      42
DEQATN     200      F(R1,R2) = R2/R1
$ 
ENDDATA

```

A final comment on the WEIGHT response is to reiterate the caution on [Defining the Objective Function](#) that, since the WEIGHT response gives the weight of the entire structure, it may be insensitive to design



variable changes. Subtracting out the weight of the undesigned structure using a DRESP2 may improve the optimizer performance. Alternatively, DOPTPRM parameter OBJMOD can be used to create an objective that is relative to the initial objective.

VOLUME

The VOLUME response provides the total volume of all of the structural elements in the structure based on the geometry of your model. The response can be for the entire structure or can be for an individual superelement specified in the ATT1 field. If ATT1 is blank, the total weight of the structure is used for the response. If it is zero, the weight of the residual is used.

FRMASS

The FRMASS response indicates a target mass fraction that typically is used as a constraint in a topology optimization task; (e.g., minimize the compliance of the structure while limiting the mass to 40% of the mass of the original structure). This response is only available with topology optimization.

COMP

The COMP response permits the specification of a compliance value as a design response. Compliance is simply the product of the displacement times the applied load and is typically used as an objective in a topology optimization design task to maximize structural stiffness in a static design problem.

EIGN

The EIGN response specifies an eigenvalue from a normal modes analysis as the response. The FREQ response is also associated with an eigenvalue, but the FREQ response quantity is in units of Hz. while the EIGN response is applied directly to the eigenvalue.

$$r_{freq} = \frac{1}{2\pi} \sqrt{r_{eign}}$$

The ATTB field of this response can be used to select the approximation technique that is applied when APRCOD=2 is used. By default, this approximation is performed using the Rayleigh Quotient approximation of [Equation \(2-138\)](#). If the ATTB is set to 1, the direct method is used to approximate the eigenvalue while ATTB=2 is used to specify the inverse variable approximation. The default is felt to be adequate for all cases, with the additional options available for experimentation.

CEIG

The CEIG response selects a component of a complex eigenvalue. The application for this response is in areas such as brake squeal or landing gear stability where typically there is a desire to ensure that the real part of the eigenvalue is positive. The ATTB field is used to select the component by specifying ALPHA (real part) or OMEGA (imaginary part) with ALPHA being the default. If the physical constraint is that the ALPHA value must be greater than zero, it is recommended that a DRESP2 be used to offset the DRESP1 response by some quantity in order to avoid a poorly scaled constraint.



FREQ

The FREQ response is similar to the EIGN response and the comments made above for that response apply here as well. Specification of design constraints in Hz is considered as being more intuitive in the typical application. The ATTB options of the EIGN response are applicable here as well and the default Rayleigh Quotient Approximation is again considered best.

LAMA

The LAMA response refers to a buckling eigenvalue and indicates what multiple of the applied load in a statics analysis would cause the structure to buckle. Typically then, the response is constrained to be greater than 1.0. The ATTA field specifies the buckling mode of interest and it is recommended that more than one LAMA response be applied to preclude buckling behavior. If only a single LAMA response is constrained, it is likely that the optimization will produce a design where a previously non-critical mode becomes critical. The ATTB field can again be used to specify the approximation technique. In this case, the Rayleigh Quotient does not apply. Instead, ATTB=1 (default) selects the direct method while ATTB=2 select the inverse variable approximation.

DISP

The DISP response identifies a displacement in a statics or static aeroelastic analysis. The ATTA field specifies the grid component(s) while the ATTi fields specify the grids at which the response is to be evaluated. The ATTA field can be any combination of the digits 1-6 (e.g., 135) and each digit will result in the recovery of a displacement (e.g., components 1,3 and 5 when ATTA is 135). The ATTB field is used for ANALYSIS = MODES subcases to identify the mode number. ATTB is ignored for ANALYSIS = STATICS. Any number of grids can be specified using the ATTi fields. The grids must exist and there is no support for a THRU option.

STRAIN

The STRAIN response invokes a strain response on a finite element. In this case, the ATTA field selects the strain item code and the meaning of the ATTi field is dependent on what has been input in the PTYPE field of the entry. If the PTYPE is "ELEM", then the ATTi data select the element IDs for the response (MSC Nastran requires that all element IDs be unique across element types so that there is no ambiguity when selecting the element IDs.) If the PTYPE selects a property type, such as PROD or PSHELL, the ATTi field selects property IDs of the type selected with PTYPE. The ATTB field is used for ANALYSIS = MODES subcases to identify the mode number. ATTB is ignored for ANALYSIS = STATICS.



Item codes for stress and strain can be found in [Item Codes](#) (p. 1023) in the *MSC Nastran Quick Reference Guide*. An example using the CQUAD4 element type with center strains is shown in [Table 4-6](#).

Table 4-6 Item Code Example for the QUAD4 Element with Linear Output

Element Name (Code)	Real Stresses or Strains		Complex Stresses or Strains		
	Item Code	Item	Item Code	Item	Real/Mag. or Imag./Phase
CQUAD4 (33) Linear	2	Z1=Fiber distance 1	2	Z1=Fiber distance 1	
	3 ¹	Normal x at Z1	3 ¹	Normal x at Z1	RM
	4 ¹	Normal y at Z1	4 ¹	Normal x at Z1	IP
	5 ¹	Shear xy at Z1	5 ¹	Normal y at Z1	RM
	6	Shear angle at Z1	6 ¹	Normal y at Z1	IP
	7	Major principal at Z1	7 ¹	Shear xy at Z1	RM
	8	Minor principal at Z1	8 ¹	Shear xy at Z1	IP
	9	von Mises or maximum shear at Z1	9	Z2=Fiber distance 2	
	10	Z2=Fiber distance 2	10 ¹	Normal x at Z2	RM
	11 ¹	Normal x at Z2	11 ¹	Normal x at Z2	IP
	12 ¹	Normal y at Z2	12 ¹	Normal y at Z2	RM
	13 ¹	Shear xy at Z2	13 ¹	Normal y at Z2	IP
	14	Shear angle at Z2	14	Shear xy at Z2	RM
	15	Major principal at Z2	15	Shear xy at Z2	IP
	16	Minor principal at Z2			
	17	von Mises or maximum shear at Z2			

Note: This table is applicable when a STRAIN Case Control command is absent in case control or present with the CENTER (default) option. As an example from the table, it is seen that item code 7 selects the major principal stress at the center Z1 location.

ESE

The ESE response type identifies an element strain energy response. The ATTB field is used for ANALYSIS = MODES subcases to identify the mode number. ATTB is ignored for ANALYSIS = STATICS. The ATTA field selects the element strain energy item code and the ATTi field selects the applicable elements or properties in the same manner as just described for the STRAIN



response. The item codes are found in [Element Strain Energy Item Codes](#) (p. 1079) in the *MSC Nastran Quick Reference Guide*. Only item codes 2 and 4 from this table are selectable on a DRESP1 entry and it is noted from the reference that not all element types are supported with item code 4. The sensitivity of the ESE response is not computed for design variables which affect the shape of the model.

STRESS

The STRESS response invokes a stress response in an element. The comments for the STRAIN response apply to the STRESS response.

FORCE

The FORCE response invokes a force response in an element. The ATTA field in this case points to item codes contained in [Element Force Item Codes](#) (p. 1060) in the *MSC Nastran Quick Reference Guide*. It is noted from [Table 4-5](#) that the direct form of approximation is used for the response type. In general, the force in an element is a weak function of the model properties making it difficult to satisfy constraints imposed on this response. The ATTB field is used for ANALYSIS = MODES subcases to identify the mode number. ATTB is ignored for ANALYSIS = STATICS.

Fatigue

The FATIGUE response is available for ANALYSIS=STATICS subcases. The format of the input is similar to other static element responses such as stress in that the PTYPE field specifies whether the ATTi refers to a particular element or to a property ID. Unlike other static element responses, the DRESP1 must point to an element that has participated in the fatigue analysis. The ATTA field points to item codes as defined in [Fatigue Item Codes, 1080](#) in the *MSC Nastran Quick Reference guide*. Available item codes are 4 -9 and 12 for the first node of the element or the element center if LOC=ELEM has been used on the FTGPARM entry. For subsequent nodes, the item code needs to be incremented as specified in [Fatigue Item Codes, 1080](#). The ATTB field must be set to the FATIGUE ID that has been invoked in case control. Note that while multiple fatigue analyses are supported in a given submittal, optimization supports only a single fatigue analysis so that the same FATIGUE ID must be used for all the DRESP1 entries with TYPE=FATIGUE. As discussed above, the meaning of the ATTi field input is dependent on the PTYPE data. If ATT1 is blank, all the fatigue responses that match the property id are extracted. Due to the fact that the fatigue analysis is likely to span subcases, the DESGLB case control command should be used to apply the constraints above the subcase level. For more details see [Design Optimization](#) (Ch. 9) in the *MSC Nastran Embedded Fatigue User's Guide*.

SPCFORCE

The SPCFORCE response invokes a single point constraint response at a designated degree-of-freedom in the model. The ATTA and ATTi field attributes follow the rules of the DISP response.

CSTRAIN

The CSTRAIN response invokes a strain response in a finite element whose properties have been provided on a PCOMP or PCOMPG entry. The item codes for composite properties differ so you must make sure you are using the correct element code when requesting composite responses. Note that corner stresses are not available for composite elements. Also, composite responses differ from homogeneous element responses in that the von Mises stress/strain is not available. The composite responses are



derived on a layer by layer basis and the ATTB field is used to identify which layer is of interest. This number corresponds to the value of i in the T_i input on the PCOMP. For the PCOMPG, ATTB points to the GPLYID i . ATTB cannot be blank on PCOMPG references. If the ATTB field is left blank for the PCOMP, the response will be calculated for the first layer.

It is noted that it is possible use the PCOMP entry but still request STRAIN/STRESS type responses for the elements by either using PTYPE = PSHELL or by identifying the elements by their ID.

CSTRESS

The CSTRESS response type is the composite stress equivalent to the CSTRAIN response for composite strains. The remarks given for the CSTRAIN response type apply equally to the CSTRESS type.

CFailure

The CFailure response type allows the specification of a composite failure criterion in the design of composite structures. The ATTB and ATT i fields for this response are the same as the CSTRAIN or CSTRESS, but the item codes differ in that the failure item codes are provide in [Element Force Item Codes](#) (p. 1060) in the *MSC Nastran Quick Reference Guide*. Only item codes 5 (direct stresses) or 7 (interlaminar shear stress) are supported. Further, the specification of the failure criterion depends on correctly specifying a number of other items in the bulk data file:

- The failure theory to be used (e.g., Tsai-Wu) is selected on the FT field of the PCOMP or PCOMPG entry.
- The allowable bonding shear stress is taken from the SB field on the PCOMP or PCOMPG entry.
- Stress limits that are used in computing the failure index are taken from the ST,SC and SS fields on the MAT i entry invoked by the PCOMP or PCOMPG.

CSTRAT

The composite strength ratio response is a direct failure indicator and, as such, is an ideal response for the design of composites. Only item codes 5 and 7 from the composite element force item codes are available for this response type.

TOTSE

The TOTSE response is a global response quantity in that the entire structure contributes to the total strain energy response due to a static load in an ANALYSIS = STATICS subcase or a normal mode in an ANALYSIS = MODES subcase. The ATTB field is used to identify the mode number in a normal modes analysis. The response can be for the entire structure or can be for an individual superelement specified in the ATT1 field. If ATT1 is blank, the total strain energy of the structure is used for the response. If it is zero, the total strain energy of the residual is used.

GPFORCE

The GPFORCE response type allows the specification of a grid point force response in a statics analysis. The ATTA field can be any combination of the digits 1-6 (e.g., 135) and each digit will result in the recovery of a grid point force (e.g., components 1, 3, and 5). Any number of elements can be specified



using the ATTi fields. The PTYPE field is used in this case to indicate which grid the grid point force is associated with.

For standard data recovery in MSC Nastran, setting NOELOF > 0 allows for a special type of grid point force printout that is along the edges of the elements. This form of grid point force is not supported on the DRESP1 entry.

GPFORCP

The GPFORCP response type allows the specification of a grid point force response that is of the form output when the PARAM NOELOP > 1 is used for standard data recovery. In this form, the sum of the grid point forces are computed parallel to the edges of the adjacent elements. The ATTA field identifies the grid point and the ATTi field(s) identify the orientation of the force by specifying the grid ID(s) that is connected to the ATTA grid.

FRDISP

The FRDISP response type identifies an displacement in a frequency response analysis at a particular grid point and component. It is therefore similar to the DISP response in statics except that the component specified by ATTA can now range from 1 through 12, where components 7-12 refer to the imaginary part of the response or to the phase angle. It is no longer possible therefore to specify multiple components using the ATTA field. As indicated in the Case Control description of [Case Control Section](#), the selection of whether the REAL/IMAGINARY or MAGNITUDE/PHASE representation is to be used is based on the option selected on the DISPLACEMENT Case Control command. In the absence of this command, the DRESP1 response is based on the default of the REAL/IMAGINARY option. Multiple subcases are supported for frequency response analysis and the set of excitation frequencies, as well as design responses and constraints, can be changed between subcases. The ATTi fields are used to identify the grid points the responses are to be extracted from while the ATTB field has a number of options that need to be covered in detail.

If the ATTB field is blank, the response is evaluated at all the excitation frequencies. As indicated on the DCONSTR entry, it is possible to specify a range of frequencies over which the constraint is applied. In this case, the constraints will only be evaluated over a range of the frequencies. If the excitation frequencies are based on FREQ3, FREQ4 and/or FREQ5 Bulk Data entries, the excitation frequencies change as the design changes so that the frequencies at which the DRESP1 responses are evaluated will change as well.

If the ATTB field is a real number, the response is calculated at the excitation frequency (in Hz.) that is nearest to that number.

ATTB field can also be one of six special character strings that have the following meaning:

$$\text{SUM} = \sum_{i=1}^n X_i$$

$$\text{AVG} = \sum_{i=1}^n X_i / n$$



$$\text{SSQ} = \sum_{i=1}^n X_i^2$$

$$\text{RSS} = \sqrt{\sum_{i=1}^n X_i^2}$$

MAX = Largest value among X_i (i=1 to n)

MIN = Smallest value among X_i (i=1 to n)

The x_i is the response at a particular frequency identified by the i subscript and the n is the number of excitation frequencies in the analysis. These special commands provide a convenient alternative to the DRESP2/DEQATN formulation and are similar to the specialized intrinsic functions discussed with the DEQATN and FUNC attribute on the DRESP2 entry. A difference is that the use of the ATTB character string on the DRESP1 applies the function to the particular frequency response results. The FUNC attribute on the DRESP2 entry is more general in that it applies to all the arguments while the special DEQATN functions are even more general because the arguments are explicitly selected.

A special situation exists when the result from a DRESP1 that includes character string input in the ATTB field is subsequently used in a DRESP2 entry. The DRESP1 in this case is regarded as a DRESP2 so that the DRESP1 ID should be referenced as a DRESP2 on the subsequent DRESP2. The following bulk data fragment shows how two DRESP1 entries that provide a RSS response at a particular grid can be summed into a single response.

```

FREQ1      100      0.0      1.0      100
$ DRESP1    410 PROVIDES A SUM OF SQUARES OF THE ACCELERATION OF COMPONENT 2
OF GRID 100
$ OVER THE 101 FREQUENCY POINTS FROM 0.0 TO 100. HZ.
DRESP1    410      ACEL1      FRACCL      2      SSQ      100
$ DRESP1    420 PERFORMS THE SAME FUNCTION FOR GRID 200
DRESP1    420      ACEL2      FRACCL      2      SSQ      200
$ DRESP2 501 USES THE "DRESP2" FLAG TO SUM THE TWO RESPONSES THAT ARE ACTUALLY
DRESP1'S
DRESP2      501          SUM
DRESP2      410          420

```

PRES

The PRES response identifies an acoustic pressure that is computed in a frequency response analysis. These pressures are computed for grid points that are in the fluid of the model and output as component 1 of that grid. The ATTA and ATTi fields are similar to the other frequency response types except that the ATTA field should only specify component 1 or 7. The DISP Case Control command selects whether the REAL/IMAGINARY or the MAGNITUDE/PHASE representation of data is used (input on the PRESSURE Case Control command is ignored). The comments regarding the ATTB field for the FRDISP response type apply to the PRES response type as well.

FRVELO

The FRVELO response identifies a velocity in a frequency response analysis at a particular grid point and component. The comments for the FRDISP response type apply to the FRVELO response type. The



VELO Case Control command selects whether the REAL/IMAGINARY or the MAGNITUDE/PHASE representation of data is used.

FRACCL

The FRACCL response identifies an acceleration in a frequency response analysis at a particular grid point and component. The comments for the FRDISP response type apply to the FRACCL response type. The ACCE Case Control command selects whether the REAL/IMAGINARY or the MAGNITUDE/PHASE representation of data is used.

FRFORC

The FRFORC response identifies an element force in a frequency response analysis. The comments regarding the ATTB field for the FRDISP response type apply to the FRFORC response type as well. The FORCE Case Control command selects whether the REAL/IMAGINARY or the MAGNITUDE/PHASE representation of data is used.

The ATTA and ATTi fields are similar to that described for the FORCE response. In selecting items codes from [Element Stress \(or Strain\) Item Codes](#) (p. 1025) in the *MSC Nastran Quick Reference Guide*, make sure that the Complex Element Force columns are used.

FRSTRE

The FRSTRE response identifies an element stress in a frequency response analysis. The comments regarding the ATTB field for the FRDISP response type apply to the FRSTRE response type as well. The STRESS Case Control command selects whether the REAL/IMAGINARY or the MAGNITUDE/PHASE representation of data is used.

The ATTA and ATTi fields are similar to that described for the STRAIN response. In selecting items codes from [Element Stress \(or Strain\) Item Codes](#) (p. 1025) in the *MSC Nastran Quick Reference Guide*, make sure that the Complex Element STRESS columns are used.

FRSPCF

The FRSPCF response identifies an single point constraint force in a frequency response analysis. The comments for the FRDISP response type apply to the FRSPCF response type. The SPCF Case Control command selects whether the REAL/IMAGINARY or the MAGNITUDE/PHASE representation of data is used. A comment on the FRSPCF response is that even though it appears to be a grid response, it is not treated as such in the determination of whether adjoint sensitivity analysis is appropriate (see [Design Sensitivity Analysis](#)). This is because the required df/du adjoint load vector of [Equation \(2-85\)](#) is not trivial to construct for single point constraint forces.

PSDDISP

The PSDDISP response identifies a power spectral density displacement value that is generated in a frequency response analysis. The form of this response is shown in [Equation \(2-18\)](#). The use of the ATTA, ATTB and ATTi fields are the same as the FRDISP. In this case, the REAL/IMAGINARY form of the data is used to construct the PSD value so the Case Control command does not play a role in selecting the data representation. The PTYPE field identifies the ID of the RANDPS Bulk Data entry that is used to define the input power spectrum. The use of the RANDPS enables this response to span



subcases. If a PSDDISP response is being applied with RANDPS Bulk Data entries that include multiple subcases, the DESSUB or DESOBJ request that invokes the PSDDISP response must be in the first subcase used by the RANDPS entry.

PSDVELO

The PSDVELO response identifies a Power Spectral Density velocity value that is generated in a frequency response analysis. The comments for the PSDDISP response are applicable with the PSDVELO response as well.

PSDACCL

The PSDACCL response identifies a Power Spectral Density acceleration value that is generated in a frequency response analysis. The comments for the PSDDISP response are applicable with the PSDACCL response as well.

RMSDISP

The RMSDISP response identifies a root mean square displacement value that is generated in a frequency response analysis. The form of this response is shown in [Equation \(2-20\)](#). The use of the ATTA, ATTB and ATTi fields are the same as the FRDISP. In this case, the REAL/IMAGINARY form of the data is used to construct the PSD value so the Case Control command does not play a role in selecting the data representation. The ATTB field identifies the ID of the RANDPS Bulk Data entry that is used to define the input power spectrum. The use of the RANDPS enables this response to span subcases. If a RMSDISP response is being applied with RANDPS Bulk Data entries that include multiple subcases, the DESSUB or DESOBJ request that invokes the PSDDISP response must be in the first subcase used by the RANDPS entry.

RMSVELO

The RMSVELO response identifies a root mean square velocity value that is generated in a frequency response analysis. The comments for the RMSDISP response are applicable with the RMSVELO response as well.

RMSACCL

The RMSACCL response identifies a root mean square acceleration value that is generated in a frequency response analysis. The comments for the RMSDISP response are applicable with the RMSACCL response as well.

ACPWR

The ACPWR response identifies the acoustic power radiated through wetted surface and all panels or through a specified panel. The output is requested by Case Control command ACPOWER. The PTYPE should have the panel name for acoustic power through panel. If PTYPE is blank, the total acoustic power radiated for all panels is calculated. ATTA and ATTi fields are blank. Field ATTB can specify the forcing frequency. The default of blank specifies all forcing frequencies.



ACINTS

The ACINTS response identifies acoustic intensity normal to wetted surface. The output is requested by Case Control command INTENSITY. The PTYPE and ATTA fields should be left blank. Field ATTB should specify forcing frequency. The default is blank for all forcing frequencies. Field ATTi should contain the Grid ID of wetted surface.

AFPRES

The AFPRES response identifies acoustic pressure for Acoustic Field Point Mesh (AFPM). The output is requested by Case Control command ACFPMRESULT. The PTYPE contains the AFPM ID. ATTA field should only specify component 1 or 7. Field ATTB should specify forcing frequency. The default is blank for all forcing frequencies. ATTi field should specify the Grid ID of AFPM.

AFINTS

The AFINTS response identifies acoustic intensity for Acoustic Field Point Mesh (AFPM). The output is requested by Case Control command ACFPMRESULT. The PTYPE contains the AFPM ID. ATTA field should specify component code - 0 for normal to AFPM and 1, 2 or 3 for x-, y- or z-directions respectively. Field ATTB should specify forcing frequency. The default is blank for all forcing frequencies. ATTi field should specify the Grid ID of AFPM.

AFVELO

The AFVELO response identifies particle velocities for requested Acoustic Field Point Mesh (AFPM) grid points. The output is requested by Case Control command ACFPMRESULT. The PTYPE contains the AFPM ID. ATTA field should specify component code - 11, 12 or 13 for real (magnitude) component of velocity respectively, or 71, 72 or 73 for imaginary (phase) component of velocity respectively. Field ATTB should specify forcing frequency. The default is blank for all forcing frequencies. ATTi field should specify the Grid ID of AFPM.

AFPWR

The AFPWR response identifies acoustic power for various frequencies through Acoustic Field Point Mesh (AFPM). The output is requested by Case Control command ACFPMRESULT. The PTYPE contains the AFPM ID. ATTA and ATTi fields are blank. Field ATTB should specify forcing frequency. The default is blank for all forcing frequencies. ATTi field should specify the Grid ID of AFPM.

Equivalent Radiated Power (ERP)

Equivalent Radiated Power identifies the amount of acoustic power being radiated into a cavity. These responses can be designed within a frequency response analysis. The item code of the ATTA field can be

2 - ERP value

3 - ERP fraction (i.e., the power of the identified panel divided by the total power).

4 - ERP in Db.

Since the response is used in frequency response, the comments regarding the ATTB field for the FRDISP response type apply to the ERP response as well. The ATTi fields can point to a specific SET3



ID, or, if left blank, all panel responses identified in the analysis will be used. The PTYPE field needs to be specified as ERPPNL.

TDISP

The TDISP response identifies a displacement in a transient response analysis at a particular grid point and component. It is therefore similar to the DISP response in statics, including the feature that it possible to specify multiple components using the ATTA field. The ATTi fields are used to identify the grid points the response is to be extracted from while the ATTB field has a number of options that need to be covered in detail.

If the ATTB field is blank, the response is evaluated at all the excitation time steps.

If the ATTB field is a real number, the response is calculated at the excitation time step that is nearest to that number.

ATTB field can also be one of six special character string discussed with the FRDISP response except that now the operations are occurring over multiple time steps instead of multiple frequencies.

TVELO

The TVELO response identifies a velocity in a transient response analysis at a particular grid point and component. The comments for the TDISP response type apply to the TVELO response type.

TACCL

The TACCL response identifies an acceleration in a transient response analysis at a particular grid point and component. The comments for the TDISP response type apply to the TACCL response type.

TSPCF

The TSPCF response identifies an single point constraint force in a transient response analysis at a particular grid point and component. The comments for the TDISP response type apply to the TSPCF response type.

TSTRE

The TSTRE response identifies an element stress in a transient response analysis. In this case the ATTA field refers to a real stress item code as given in [Element Stress \(or Strain\) Item Codes](#) (p. 1025) in the *MSC Nastran Quick Reference Guide*. The comments on the ATTB field provided with the TDISP description are applicable to the TSTRE response as well.

TFORCE

The TFORCE response identifies an element force in a transient response analysis. In this case the ATTA field refers to a real force item code as given in [Element Force Item Codes](#) (p. 1060) in the *MSC Nastran Quick Reference Guide*. The comments on the ATTB field provided with the TDISP description are applicable to the TFORCE response as well.



STMONP1

The STMONP1 response invokes a structural MONPNT1 (as opposed to an aerodynamic MONPNT1). This response is only available in static aeroelastic (ANALYSIS=SAERO) and static (ANALYSIS=STATICS) subcases and not in dynamic or modal subcases. Invoking this response in a statics subcase may not make much sense since loads are typically invariant with respect to the designed properties. The ATTA field specifies the components to be extracted. These can be any subset of the integers 1 through 6 that appear on the monitor quantity with the NAME provided in the PTYPE field.

STMOND1

The STMOND1 response invokes a structural MONDSP1 (as opposed to an aerodynamic MONDSP1). This response is only available in static aeroelastic (ANALYSIS=SAERO) and static (ANALYSIS=STATICS) subcases and not in dynamic or modal subcases. The ATTA field specifies the components to be extracted. These can be any subset of the integers 1 through 6 that appear on the monitor quantity with the NAME provided in the PTYPE field. This response can be thought of as being an averaged displacement.

MONPNT3

The MONPNT3 response invokes a MONPNT3 bulk data entry. This response is only available in static aeroelastic (ANALYSIS=SAERO) and static (ANALYSIS=STATICS) subcases and not in dynamic or modal subcases. Invoking this response in a statics subcase may not make much sense since loads are typically invariant with respect to the designed properties. The ATTA field specifies the components to be extracted. These can be any subset of the integers 1 through 6 that appear on the monitor quantity with the NAME provided in the PTYPE field.

AEMONP1

The AEMONP1 response invokes an aerodynamic MONPNT1 bulk data entry. This response is only available in static aeroelastic (ANALYSIS=SAERO) subcases. The ATTA field specifies the components to be extracted. These can be any subset of the integers 1 through 6 that appear on the monitor quantity with the NAME provided in the PTYPE field.

AEMOND1

The AEMOND1 response invokes an aerodynamic MONDSP1 bulk data entry. This response is only available in static aeroelastic (ANALYSIS=SAERO) subcases. The ATTA field specifies the components to be extracted. These can be any subset of the integers 1 through 6 that appear on the monitor quantity with the NAME provided in the PTYPE field. This response can be thought of as being an averaged displacement.

TRIM

The TRIM response identifies a trim variable in a static aeroelastic response analysis. This can be used to impose limits on the values these variables can take on during the analysis. It is only necessary to identify the ID of the AESTAT or AESURF entry that defines the trim variable using the ATTA field. The *MSC Nastran User's Guide for Aeroelastic Analysis, V68* provides further information on this response in [Aeroelastic Design Sensitivity and Optimization](#).



STABDER

The STABDER response identifies a stability derivative in a static aeroelastic response analysis. This can be used to impose limits on the values these variables can take on during the analysis. In addition to the AESTAT/AESURF entry in the ATTA field, a single component (1 through 6) is specified using the ATT1 field. The ATTB field is used to specify whether restrained (ATTB=1) or unrestrained (ATTB=0) form of the derivative is to be used. The *MSC Nastran User's Guide for Aeroelastic Analysis, V68* provides further information on this response in [Aeroelastic Design Sensitivity and Optimization](#) and examples in Chapter 10 of that guide show how the stability derivative for the roll rate due to the a control surface deflection can be combined with the stability derivative for the roll rate due to roll to provide a measure of the achievable roll rate of the vehicle.

FLUTTER

The FLUTTER response identifies a flutter damping value from an aeroelastic flutter analysis. In this case, the ATTi fields are used to uniquely identify the damping values of interest.

- ATT1 identifies a SET1 Bulk Data entry that specifies for which modes the responses are computed. This allows you to pinpoint the flutter roots of interest and to avoid placing constraints on flutter eigenvalues that are primarily due to rigid body modes and therefore not affected by changes in the structure.
- ATT2 identifies an FLFACT entry that specifies the densities at which the flutter responses are desired. This should be the same set or a subset of the densities used in the flutter analysis that have been selected on the FLUTTER Bulk Data entry.
- ATT3 identifies an FLFACT entry that specifies the Mach numbers at which the flutter responses are desired. This should be the same set or a subset of the Mach numbers used in the flutter analysis that have been selected on the FLUTTER Bulk Data entry.
- ATT4 identifies an FLFACT entry that specifies the velocities at which the flutter responses are desired. Note that only the **PK** and **PKNL** methods of flutter analysis are supported for this response. This should be the same set or a subset of the velocities used in the flutter analysis that have been selected on the FLUTTER Bulk Data entry.

If the PKNL method has been specified on the FLUTTER Bulk Data entry, it must be specified in the PTYPE field of the DRESP1 entry.

DIVERG

The DIVERG response is available for the special ANALYSIS=DIVERG subcase that performs a static aeroelastic divergence analysis and the response value is the dynamic pressure of the selected root. The ATT1 field identifies the single Mach number of the divergence analysis while the ATTA field selects the divergence root of interest. Typically this would be the first root, but as with the LAMA response, it may be necessary to specify multiple roots to avoid the situation where satisfying the limit on the first divergence behavior results in a mechanism from the second or higher root falling below the desired dynamic pressure.



WMPID

The WMPID response can be used to create a response that represents the contribution to the weight of the model from a particular material or property ID. ATTA specifies the material ID and is always required. The optional ATTi entries specify the property id's associated with the response. If these PID's are specified, it is necessary to specify the property type on the PTYPE field, ATTB is another optional input that specifies the superelement ID. If ATTB is blank, the response applies to all superelements. If it is zero, it applies to the residual and if it is a positive integer, it refers to the weight associated with that particular superelement. Composite materials are supported for this response type and this includes the ability to segregate the weight of a material that is used in one or more, but not all the laminae.



Optimization Parameters

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
PARAM <i>i</i>	Name of the design optimization parameter. For allowable names, see the DOPTPRM listing in The Bulk Data Section (p. 1097) in the <i>MSC Nastran Quick Reference Guide</i> . (Character)
VAL <i>i</i>	Value of the parameter. (Real or Integer; see the DOPTPRM listing in The Bulk Data Section (p. 1097) in the <i>MSC Nastran Quick Reference Guide</i>)

Associated Entries:

None.

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 may be changed using the DOPTPRM entry. An overview of *selected* parameters, grouped according to their functionality is presented in this section. See also [The Bulk Data Section](#) (p. 1097) in the *MSC Nastran Quick Reference Guide* as it contains the complete list of all optimization parameters that may be changed.

Choice of Approximation Method (APRCOD)

There are three types of approximation methods to choose from: direct linearization, mixed method, and convex linearization (see [Function Evaluation](#)). 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 (see [Table 4-5](#)).



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 constraint basis.

Optimization Method (OPTCOD, METHOD)

The OPTCOD parameter permits the specification of the optimization code to be used as shown in [Table 4-7](#). Note that if neither system cell number 413 or OPTCOD is specified, the code selects the best optimization algorithm to use based on problem parameters. This is usually adequate.

Table 4-7 OPTCOD Options

Name	Description, Type and Default Value	
OPTCOD	OPTCOD (Character; Default = Blank)	
	= Blank	taken from system cell number 413
	= "MSCADS"	MSCADS is used
	= "IPOPT"	IPOPT is used

The METHOD parameter selects the particular algorithm that is to be used with MSCADS as shown in [Table 4-8](#). The parameter is only relevant for MSCADS. For MSCADS, METHOD's 1, 2 and 3 refer to the Modified Method of Feasible Directions (MMFD), Sequential Linear Programming (SLP) and Sequential Quadratic Programming (SQP), respectively. Experience has shown that the MMFD algorithm is the most reliable for MSCADS.

METHOD=4 is sequential unconstrained minimization technique (SUMT) and is only available with MSCADS. METHOD=ijk provides access to a suite of MSCADS optimization techniques, with the I specifying the strategy, j the optimizer and k the one-dimensional search method as shown in [Table 4-9](#). No guidelines are provided in the use of these ijk alternatives, they are simply made available for knowledgeable users who are encountering difficulties with one of the basic methods.



Table 4-8 METHOD Options

Name	Description, Type and Default Value	
METHOD	Optimization method (Integer ≥ 0 ; Default = 0)	
	0	Automatic selection for a better performance based on number of design variables, number of constraints, number of active/violated constraints and computer memory.
	1	Modified Method of Feasible Directions for MSCADS.
	2	Sequential Linear Programming for MSCADS
	3	Sequential Quadratic Programming for MSCADS
	4	SUMT method for MSCADS
	IJK	See Table 4-9

Table 4-9 MSCADS Options

MSCADS strategy options (integer 0-9)		
	0	None - Go directly to the optimizer
	1	Sequential unconstrained minimization using the exterior penalty function method
	2	Sequential unconstrained minimization using the linear extended interior penalty function method
	3	Sequential unconstrained minimization using the quadratic extended interior penalty function method
	4	Sequential unconstrained minimization using the cubic extended interior penalty function method
	5	Augmented Lagrange multiplier method
	6	Sequential linear programming
	7	Method of centers
	8	Sequential quadratic programming
	9	Sequential convex programming
MSCADS optimizer options for J (Integer 1-5)		
	1	Fletcher-Reeves algorithm for unconstrained minimization
	2	Davidson-Fletcher-Powell (DFP) variable metric method for unconstrained minimization
	3	Broydon-Fletcher-Goldfarb-Shanno (BFGS) variable metric method for unconstrained minimization



Table 4-9 MSCADS Options (continued)

	4	Method of feasible directions for constrained minimization
	5	Modified method of feasible directions for constrained minimization
MSCADS one-dimensional search options for K (integer 1-8)		
	1	Find the minimum of an unconstrained function using the Golden Section method
	2	Find the minimum of an unconstrained function using the Golden Section method followed by polynomial interpolation
	3	Find the minimum of an unconstrained function by first finding bounds and then using polynomial interpolation
	4	Find the minimum of an unconstrained function by polynomial interpolation/extrapolation without first finding bounds on the solution
	5	Find the minimum of a constrained function using the Golden Section method
	6	Find the minimum of a constrained function using the Golden Section method followed by polynomial interpolation
	7	Find the minimum of a constrained function by first finding bounds and then using polynomial interpolation
	8	Find the minimum of a constrained function by polynomial interpolation/extrapolation without first finding bounds on the solution

Transformation of the Approximate Optimization Task to a Feasible Design (PENAL)

The PENAL parameter can be used to transform an infeasible design into a feasible one using the equations

$$g_j^i = g_j - B \quad (j = 1, 2, \dots, ncon)$$

$$\Phi^1 = \Phi + B \cdot PENAL \cdot \Phi_0$$

where the algorithm initializes B to a value equal to the maximum value of g_j (if this is a negative number, the transformation is not required). Φ_0 is the initial value of the objective.

This parameter has been shown to be most useful when applied with the ADS algorithms. $PENAL=2.0$ is recommended.



Design Cycle Print Controls (P1, P2, P2CR, P2CDDV, P2CP, P2CC, P2CM, P2CBL, P2CALL, P2RSET)

Insight into the progress of the optimization task can be obtained by requesting prints of selected information during the optimization. This can potentially produce an unmanageable amount of data so that a number of DOPTPRM parameters are available for specifying this print. P1 controls the frequency of the output:

- P1 = 0 output for initial and optimal designs (default)
- = n output for every n-th design cycle

P2 provides a “first level” control of which design quantities are printed:

- P2 = 0 no output
- = 1 objective and design variables (default)
- = 2 designed properties
- = 4 design constraints
- = 8 design responses
- = 16 weight as a function of material ID

The various P2 options can be summed to produce output combinations. For example, P2 = 15 prints out all available design data. The P2 = 16 option is the result of a special request and is not directly related to design.

Eight additional parameters control constraint and response prints once these prints have initially been invoked with the P2 parameter.

P2CR	Maximum number of C onstraints on R esponses to be printed
P2CDDV	Maximum number of C onstraints on D ependent D esign V ariables to be printed
P2CP	Maximum number of C onstraints on P roperties to be printed
P2CC	Maximum number of C onstraints on C onnectivity properties to be printed
P2CM	Maximum number of C onstraints on M aterial properties to be printed
P2CBL	Maximum number of C onstraints on B eam L ibrary dimensions to be printed
P2CALL	Maximum number of C onstraints of all categories to be printed
P2RSET	SET1 ID for which a set of Response IDs is defined

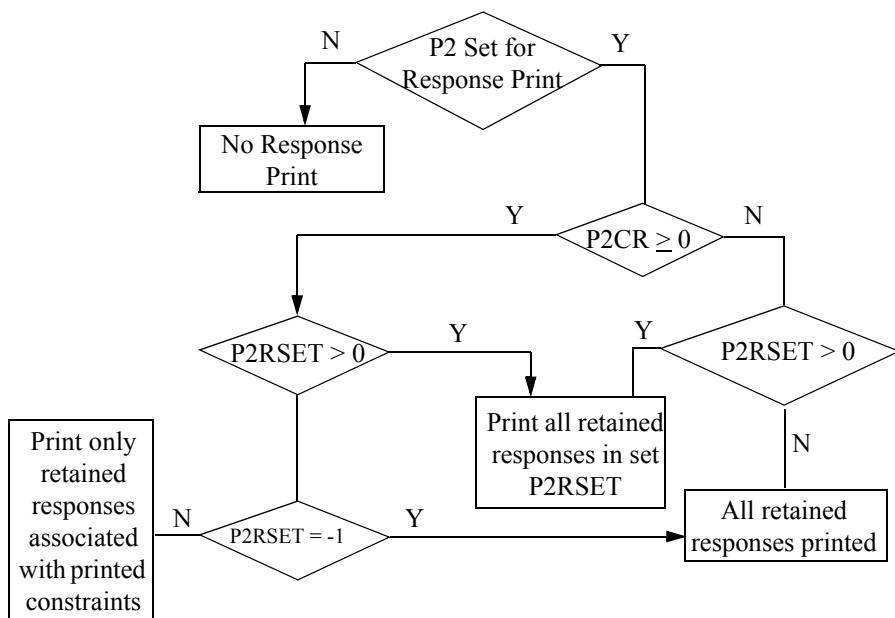
P2CALL can be used to provide a default for any or all of the categories. For example, P2CALL = 10 will print up to 10 of each type of constraint. If there are fewer than ten constraints of a give type available, only the available number will be printed. Note that P2CALL applies to each type of constraint individually. No attempt is made to find the most critical constraints across all types and only print out these constraints.



The remaining P2Cx parameters allow you to selectively specify the number of constraints printed for that constraint type. If the number specified is fewer than the number available, a sort is performed to find the P2Cx most critical and only these are printed. If the number specified exceeds the number of constraints available, no sorting is performed and all the constraints are printed in the order of increasing internal constraint ID.

P2RSET can be utilized to print a selected set of responses. P2RSET refers to a SET1 Bulk Data entry that contains a list of DRESPx IDs that are to be printed. Only retained responses (i.e., those that survived the constraint screening process) are printed.

The print of responses can now be affected by three DOPTPRM parameters: P2, P2CR and P2RSET. [Figure 4-2](#) depicts how the three parameters affect the print.



[Figure 4-2](#) Flowchart Showing How P2, P2RSET and P2CR Affect the Print of Retained Design Responses

Optimizer Level Print Control (IPRINT)

The parameter IPRINT controls the amount of optimizer output. This parameter is normally turned off (default = 0) by users since the optimizer is used in a “black-box” mode. 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

IPRINT	Optimizer Printout
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

Note that IPRINT ≥ 3 produces constraint prints. These constraint values are in the internal constraint order, which is associated with the constraint evaluation and is not affected by the print order produced with the P2Cx parameters. The internal constraint ID is still printed when the P2Cx parameters are used, but the constraints are unlikely to be in ascending order of this ID.

Maximum Number of Mathematical Programming Design Cycles (DESMAX)

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 or until the maximum specified number of design cycles (DESMAX) is reached. The default value of DESMAX is five. Usually, good progress toward a optimal design has been made by this point and it's a good time to review the results. However, more cycles are often necessary and the DESMAX parameter can be adjusted at the user's discretion. At other times, it is desirable to set DESMAX to one so that you can make sure that everything appears to be in order before committing to a longer run. If the maximum number of design cycles is reached before convergence is achieved, the problem can always be restarted from the last design (see [Topology Optimization](#)).

Fully Stressed Design (FSDMAX, FSDALP)

The Fully Stressed Design (FSD) algorithm discussed in [Fully Stressed Design](#) (Ch. 9) has a design loop that is independent from the mathematical programming loop. The maximum number of FSD design cycles is specified by FSDMAX, which has a default value of 0 (no FSD cycles). Once the FSD cycles have been completed (either from convergence or after FSDMAX design cycles), the algorithm automatically transitions to the mathematical programming algorithm with DESMAX used to control the maximum number of additional cycles that will be made. Setting DESMAX to 0 will prevent any mathematical programming cycles.

The FSDALP parameter controls the relaxation factor, α , used in the resizing equation of [Equation \(9-1\)](#). The default value of 0.9 for FSDALP is usually adequate to achieve quick convergence. Lower values, such as 0.5, can be used to achieve a completely converged solution.



Move Limits on the Approximate Optimization (DELP, DPMIN, DELX, DXMIN)

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 [The Approximate Model](#), 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 and DXMIN for design variables. The DELXV attribute on the DESVAR entry allows control in the movement of an individual design variable. DELP must always be less than 1.0. A value of 1.0 or greater could allow the property to go to zero or even negative, with unpredictable results.

Convergence Criteria (CONV1, CONV2, GMAX, CONVDV, CONVPR)

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. [Tests for Convergence](#), describes the types of convergence testing as well as the convergence decision logic.

Soft convergence compares the design variables and design properties output from the approximated model optimization with these same values 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 and it can be changed with a PARAM Bulk Data entry, not with the DOPTPRM entry.

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 can also be controlled using the DOPTPRM entry. Parameters that can be changed include DABOBJ and DELOBJ.

Finite Difference Perturbations (DELB, STPCSL)

Design sensitivity coefficients are computed in MSC Nastran using a semianalytic approach, as explained in [Design Sensitivity Analysis](#).

The value of the property finite difference move parameter DELB of [Equation \(2-40\)](#) and [Equation \(2-41\)](#) can be changed with the DOPTPRM entry although the default value of 1.0E-4 gives good results in most cases.

The value of the shape finite difference move parameter STPCSL of [Equation \(2-42\)](#) can also be changed with the DOPTPRM entry.

Comparison Between the Design and Analysis Properties (PTOL, PLVIOL)

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 differs 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. If you want your analysis models and



design models to agree, setting PTOL to a value close to zero will terminate the run when the two models differ significantly. A related DOPTPRM parameter is PLVIOL, which can be set to a non-zero integer to allow the run to continue when the initial designed properties violate the limits imposed on the properties by the PMIN and PMAX values on the DVxRELY entries. Allowing these violations is not good practice since it puts a burden on the optimizer to satisfy a violated property constraint that is more properly satisfied when preparing the input.

A final comment on analysis versus design property values is that the DPEPS parameter entered with a PARAM entry allows you to specify a difference value that is used to assess whether the analysis properties are close enough to the designed properties that previously computed results using the analysis properties can be used in subsequent sensitivity analysis. The default value of DPEPS = 1.0E-4 allows for only a very small difference so that a new analysis will be performed whenever the design properties differ from the analysis properties. Setting this parameter to a larger value allows greater differences and perhaps avoids a finite element analysis. But it must be realized that the sensitivities, which are computed using the design values, are not fully consistent with the analysis values which have been calculated using the analysis values.

Discrete Variable Optimization (DISCOD, DISBEG)

As discussed in [Discrete Variable Optimization](#) (Ch. 9), a continuous optimization task can be followed by an essentially postprocessing operation which determines the values of the design variables which provides an optimal design while limiting the design variable values to a discrete set of values specified using the DDVAL entry. Two DOPTPRM parameters (DISCOD and DISBEG) control the discrete variable processing. The DISBEG parameter specifies at which mathematical programming design cycle you want to start performing discrete optimization. The default value of 0 performs discrete optimization only after the continuous optimization process is complete. The DISCOD parameter selects from one of the four discrete optimization alternative methods.

DISCOD Value	Discrete Optimization Method
1	Design of Experiments (Default)
2	Conservative Discrete Design
3	Round up to the nearest design variable
4	Round off to the nearest design variable
0	Turn off discrete variable processing

The DISCOD = 0 option facilitates ignoring DDVAL data on the DESVAR entry when a discrete variable optimization task is not desired. Options 1-4 are discussed in greater detail in [Discrete Variable Optimization](#) (Ch. 9).

Constraint Normalization (GSCAL)

[Equation \(2-10\)](#) indicates how the constraints are normalized and makes reference to a GSCAL parameter. This parameter can be modified from its default value of 0.001 using the DOPTPRM entry. As the discussion following [Equation \(2-10\)](#) indicates, it is not good practice to define constraints in a way



that requires the use of GSCAL. Instead, the response should be recast in a form (perhaps through the use of a DRESP2 entry) that avoids the small bounds that invoke the use of GSCAL.

Active and Violated Constraints (CT, CTMIN)

The MSCADS optimizer makes a determination of what constraints it should consider when performing its task that is over and above the screening actions discussed in [Constraint Screening](#). DOPTPRM parameter CT is used to define that threshold above which a constraint is considered active while CTMIN is used to identify violated constraints. (See [Figure 2-8](#) and [Figure 2-9](#).) The default values of -0.03 for CT and 0.003 for CTMIN have been found to be adequate in most cases. When it appears that the default value is causing the optimizer to neglect constraints that become critical once the redesign is performed, the CT value can be changed to a value such as -0.1.

Topology Optimization Variables (TCHECK and TDMIN) and Special Topology Optimization Settings

Topology optimization has two parameters that are unique to it. Parameter TCHECK is used to turn on a filtering parameter that is used to prevent a checkerboard pattern from being produced.

The value 1 turns on the filtering algorithm. Use TCHECK=2 to invoke Density Constraint method when the minimum member size is relatively larger. The default value -1 is automatic selection of the filtering algorithm and density constraint and is the recommended option. TCHECK=0 turns off filtering.

The second parameter, TDMIN, is used to prevent achieving a final design that is characterized by thin disjoint fibers that are impractical from a manufacturing standpoint. Design variable elements that are within a distance of TDMIN from an element with a design variable close to 1.0 are filtered to ensure they are not small. Note that the value of TDMIN is problem-size dependent but not mesh-size dependent. The default for TDMIN is 0.0 (no filtering).

Tuning of the topology optimization algorithm has indicated the following parameters benefit from a different default value than is used in standard design optimization.

Table 4-10 Modified DOPTPRM Default for Topology Optimization

Parameter	Topology Default
DESMAX	30 or 60 for Topology with minimum member size
CONVDV	1.0E-4
DELX	0.2
DXMIN	1.0E-5

DOPTPRM parameter P2 also has a different meaning for topology optimization in that design variables are printed only when $P2 > 8$. This is because topology optimization tasks typically involve thousands of design variables.



External Response

External Response or Type-3 Response

Type-3 responses provide a completely general way of synthesizing responses which are (i) not Nastran responses or Type-1 responses and (ii) too complex to evaluate using synthetic formulations or Type-2 responses.

The Type-3 response is regarded as a major extension of the Type-2 response.

The representation of Type-3 response is similar to that of Type-2. To illustrate, equations (2-7) and (2-8) from [Fundamentals of Design Sensitivity and Optimization in MSC Nastran](#) are reproduced below as equations (4-1) (for Type-2) and (4-2) (for Type-3), respectively:

$$r^2 = f(X, C, R1, P, R2, XYZ) \quad (4-1)$$

$$r^3 = f(X, C, R1, P, R2, XYZ, \text{string}) \quad (4-2)$$

However, there are some differences. Two of them can be immediately spotted from a cursory glance at the equations (4-1) and (4-2). While Type-2 responses may be functions of other Type-2 responses (note R2 in the argument list in equation 2-7), Type-3 responses cannot be functions of Type-3 responses. The additional 'string' argument in equation 2-8 permits the passing of user data that can be used to direct the execution of the program calculating the response.

A major difference is that while the evaluation of Type-2 response is accomplished by internally solving an explicit equation defined on the DEQATN Bulk Data Entry, the evaluation of the Type-3 response is by an external program with which Nastran communicates through an API (Application Program Interface). This is what gives the Type-3 response an unlimited potential to evaluate responses of any complexity. More importantly, the external program could be proprietary; Nastran only needs to know the value of the response, and not how it is calculated.

To illustrate this difference, [Figure 3-4](#) and [Figure 3-7](#) from [Developing the Design Model for Sensitivity and Optimization](#) are reproduced as [Figure 4-3](#) and [Figure 4-4](#), respectively. They depict the data structures of DRESP2 and DRESP3 (Bulk Data entries for Type-2 and Type-3 responses, respectively).



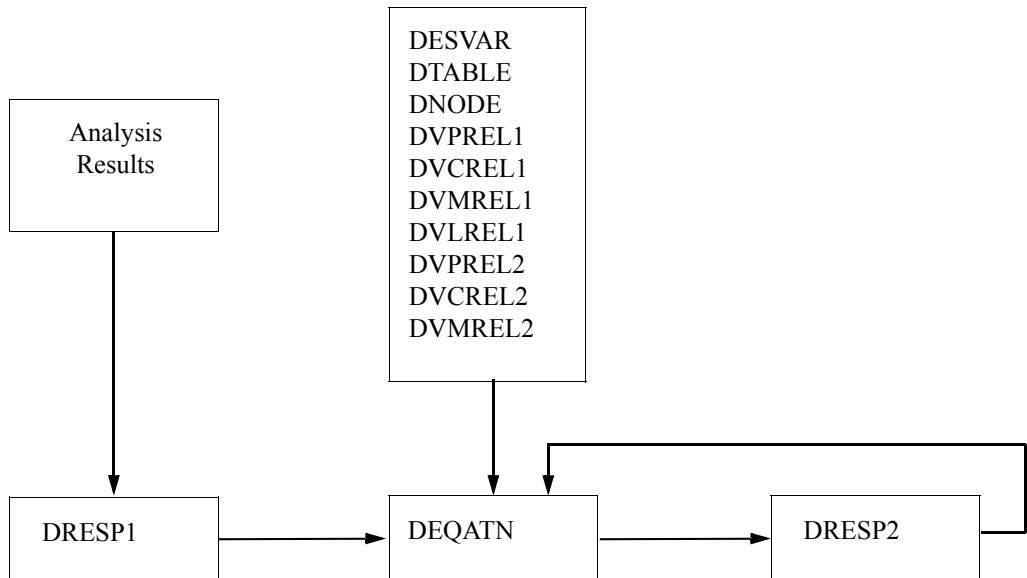


Figure 4-3 Second - Level Responses

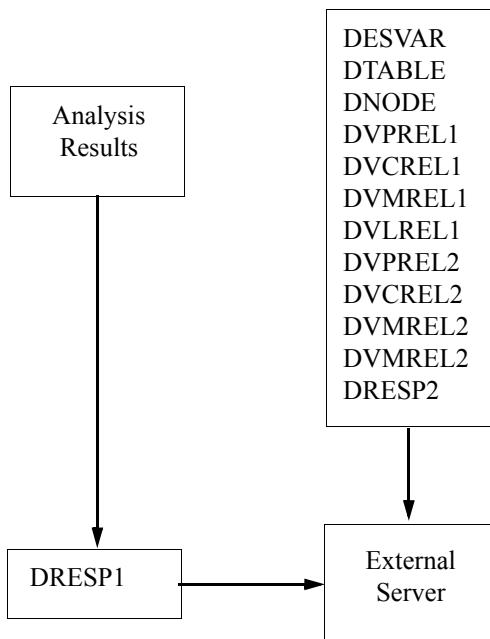


Figure 4-4 Third-Level Responses



The first line of the DRESP2 Bulk Data Entry contains field EQID or FUNC for providing the ID of DEQATN entry. DRESP3 Bulk Data Entry does not have this field. Instead, it has fields GROUP and TYPE which identify the group and type for the external response.

The client-server technology that has been used in Type-3 response is illustrated in [Figure 4-5](#).

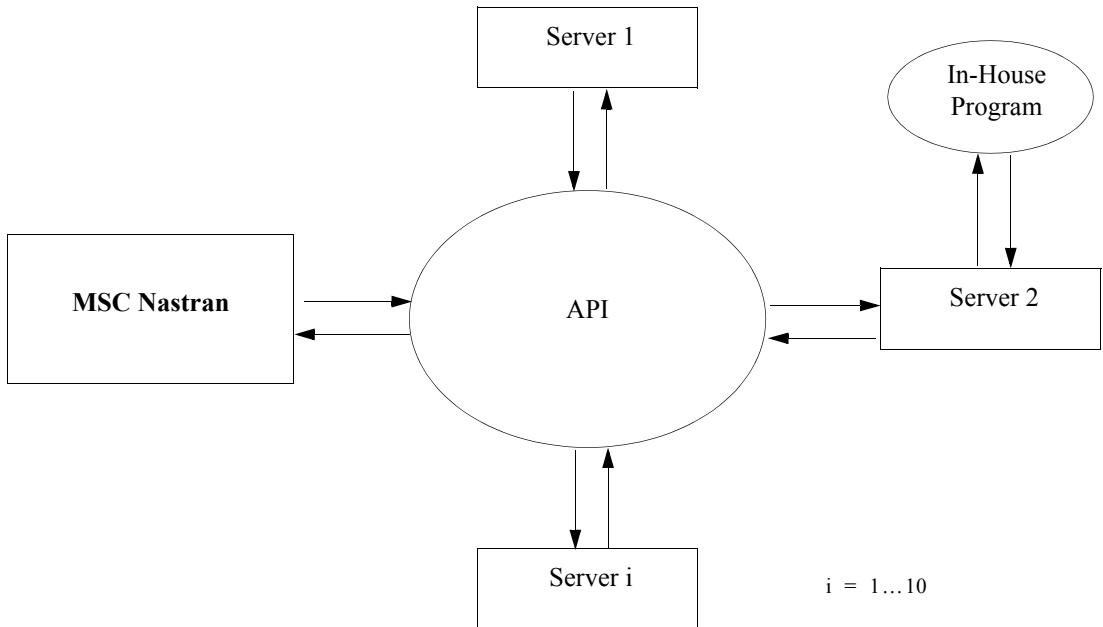


Figure 4-5 The External Response Client-Server Capabilities

User Interface for Setting Up External Response

There are six steps required in setting up the external response in a design task:

1. Create DRESP3 entry to specify the group and type of response
2. Write C or Fortran routines to calculate response based on template routines
3. Build an External server executable using the SCons Tool executable
4. Create an FMS CONNECT entry to define the external response group
5. Create an evaluator connection file to associate the external response group with the server program.
6. Submit the MSC Nastran job with the gmconn keyword that references to the evaluator connection file.

These six steps are now explained in detail.

Creating a DRESP3 entry

A DRESP3 entry defines an external response calculated using user-supplied routines. The response can be used as constraints or as an objective in a design. A compact form of the DRESP3 entry is given below. See DRESP3 for the detailed description of the Bulk Data Entry.



	1	2	3	4	5	6	7	8	9
DRESP3	ID	LABEL	GROUP	TYPE	REGION				
“DESVAR”	-----List of DESVAR IDs-----								
“DTABLE”	-----List of LABELs for constants in DTABLE entry-----								
“DRESP1”	-----List of DRESP1 IDs-----								
“DNODE”	-----List of Grid IDs and corresponding degree of freedom-----								
“DVPREL1”	-----List of DVPREL1 IDs-----								
“DVCREL1”	-----List of DVCREL1 IDs-----								
“DVMREL1”	-----List of DVMREL1 IDs-----								
“DVPREL2”	-----List of DVPREL2 IDs-----								
“DVCREL2”	-----List of DVCREL2 IDs-----								
“DVMREL2”	-----List of DVMREL2 IDs-----								
“DRESP2”	-----List of DRESP2 IDs-----								
“DVLREL1”	-----List of DVLREL1 IDs-----								
“USRDATA”	-----String-----								

Example

DRESP3	1	MYRESP	MYBUCK	BUCKEJ	
	DESVAR	1	2	3	
	DRESP1	10	11	12	
	USRDATA	Constants: 12345.6 789.0 99.0			

Write C or Fortran Routines to Calculate Response Based on Template Routines

Two subroutines are modified to perform the server task. These subroutines, and any further routines are placed in the src subdirectory described in Step 3 below. The first subroutine is entitled R3SGRT and a simple example is shown in Listing 4-1 below. This sets up the parameters used for the external response.

Listing 4-1 R3SGRT Subroutine

```
SUBROUTINE R3SGRT(GRPID,TYPNAM,nresp, grdtyp, ERROR)
C -----
C
C     PURPOSE: VERIFY THE EXTERNAL RESPONSE TYPE
C
C     GRPID    input integer      - Group ID
C     TYPNAM   input character*8   - Name of external response type
C     nresp    output integer     - number of responses for this dresp3
C     grdtyp   output integer     - integer array of length nresp
C                               indicating how gradient are to be
C                               computed
C                               = 1 user to supply analytic gradients
```



```

C           varies during approx. optimization
C           = 2 user to supply analytic gradients
C           invariant during approx. optimization
C           = 3 finite difference technique to provide gradients
C           varies during approx. optimization
C           = 4 finite difference technique to provide gradients
C           invariant during approx. optimization
C
C     ERROR      input/output integer -error code for the call.
C
C     Method
C       Match the user input: typnam with the list of available
C       external response types. If no match is found, set error code.
C
C     Called by
C           R3CGRT
C
C     NOTE:
C       The writer of this routine is responsible to specify
C       NTYPES and R3TYPE.
C -----
C
C     VARIABLES PASSED IN
C
C       INTEGER GRPID, ERROR, nresp
C       integer grdtyp(*)
C       double precision rgdtyp(*)
C       CHARACTER*8 TYPNAM
C
C     LOCAL VARIABLES
C
C       INTEGER NTYPES, BADTYP
C       PARAMETER(NTYPES=6)
C       CHARACTER*8 R3TYPE(NTYPES)
C
C       DATA BADTYP/7554/
C       DATA R3TYPE/'USEVAR1 ','USEVAR10','USEALL',
C .          'USEMIXVS','FREQMOD ','EULJOH '/
C
C       ERROR = 0
C       DO 100 ITYPE = 1, NTYPES
C         IF (TYPNAM .EQ. R3TYPE(ITYPE)) THEN
C           nresp = 2
C           grdtyp(1) = 1
C           grdtyp(2) = 1
C           GOTO 200
C         END IF
100    CONTINUE
C       ERROR = BADTYP
200    CONTINUE
C       RETURN
C     END

```

The user is required to assign a value to NTYPES in the parameter statement for the number of response types for a given response group and to assign the response type names to the array R3TYPE in the data statement. Further, the user must provide a value to the number of responses NRESP to be calculated for the response type. Finally, the user must indicate how the gradients are to be supplied for each of the response types. In the example shown above, there are two responses for the response type and analytical gradients are to be supplied.



Before listing the R3SVALD routine, a portion of a sample input deck containing the DRESP3 and related entries is given below:

DRESP3	32	JOHNSON	TESTGRP	EULJOH
		DESVAR 1		
		DTABLE L	E	SIGMAC
		DRESP1 25		
DESVAR	1	R	1.0	0.01 10.0
DRESP1	25	SIGMA	STRESS	PBAR 6 10

Note: DRESP3 has 5 arguments labeled as R, L, E, SIGMAC and S1

Listing 4-2 R3SVALD Subroutine

```

SUBROUTINE R3SVALD(GRPID,TYPNAM,
    .           NITEMS,ARGLIS,
    .           NSIZE, ARGVAL,
    .           NWRDA8,ARGCHR,
    .           forg,nresp,narg,
    .           DR3VAL,senval,
    .           error)
C -----
C
C   PURPOSE: COMPUTE THE EXTERNAL RESPONSE
C
C   GRPID   INPUT INTEGER      - GROUP ID
C   TYPNAM  INPUT CHARACTER*8 - NAME OF EXTERNAL RESPONSE TYPE
C   NITEMS  INPUT INTEGER     - DIMENSION OF ARRAY ARGLIS
C   NSIZE   INPUT INTEGER     - DIMENSION OF ARRAY ARGVAL
C   NWRDA8  INPUT INTEGER     - DIMENSION OF CHARACTER ARRAY ARGCHR
C   ARGLIS  INPUT INTEGER     - ARRAY OF NO. OF ITEMS FOR EACH
C                               ARGUMENT type
C   ARGVAL  INPUT DOUBLE      - ARRAY OF ARGUMENT VALUES (except
C                               characters)
C   ARGCHR  INPUT CHARACTER*8 - ARRAY OF CHARACTER VALUES
C   nresp   input integer     - number of responses
C   forg    input integer      - type of call
C                               = 0 function evaluation
C                               = 1 sensitivity evaluation
C   narg    input integer      - number of arguments needing gradients
C   DR3VAL  OUTPUT DOUBLE     - VALUE OF THE EXTERNAL RESPONSEs
C   senval  output double     - matrix of the sensitivity of the IRth
C                               response to the IARGth argument
C   ERROR   input/output integer -error code for the call.
C
C   Method
C       A)SET UP VARIOUS PARAMETERS FROM THE ARGUMENT LIST
C       B)if forg = 0 EVALUATE THE EXTERNAL RESPONSE BASED ON THE
C          GIVEN TYPNAM
C       C)else if forg = 1 EVALUATE THE sensitivities of the external
C          responses to the arguments that can vary for

```



```

C           the given typnam
C       D) RETURN BADTYP ERROR IF TYPNAM IS NOT MATCHED HERE.
C
C   nsize - the number of arguments or values in a dresp3 entry
C
C   NSIZE=NV+NC+NR+NNC+NDVP1+NDVP2+NDVC1+NDVC2+NDVM1+NDVM2+NRR2
C   where:
C       nv      = number of desvars      nr      = number of dtables
C       nr      = number of drespls     nnc     = number of dnode pairs
C       ndvp1   = number of dvprel1s    ndvp2   = number dvprel2s
C       ndvc1   = number of dvcrel1s    ndvc2   = number dvcrel2s
C       ndvm1   = number of dvmrells   ndvm2   = number dvmrel2s
C       nrr2    = number of dresp2s
C   narg = nsize - nc
C
C   CALLED BY
C       Various
C -----
C
C   VARIABLES PASSED IN
C
C   CHARACTER*8 TYPNAM, ARGCHR(NWRDA8)
C   integer forg , nresp
C   INTEGER GRPID, NITEMS, NSIZE, ARGLIS(NITEMS), ERROR, NWRDA8
C   DOUBLE PRECISION ARGVAL(NSIZE), DR3VAL(*), senval(nresp,*)
C
C   LOCAL VARIABLES
C
C   INTEGER BADTYP, idbg
C   DOUBLE PRECISION PI,  FAC, FACT, SLNDER
C   DOUBLE PRECISION R,L,E,SIGMA,SIGMAC, RGYRA
C
C   DATA BADTYP /7554/, badfg /7555/
C
C   PI = 3.14159
C   PI2 = PI * PI
C   nresp = 2
C
C   THE USER-SUPPLIED EQUATION TO DEFINE THE EXTERNAL RESPONSES
C   SIGMA = DRESP1, R=DESVAR, L, E AND SIGMAC = DTABLE CONSTANTS
C
C   EULER : EULER= -SIGMA * (L/ RGYRA ) **2 / (PI**2 * E)
C           RGYRA = R / 2.0
C
C   JOHNSON: JOHNSON = -SIGMA / (SIGMAC * FACTOR )
C           FACTOR = 1. - SIGMAC * (L/RGYRA)**2 /(4 * PI**2 * E)
C   ERROR = 0
C
C   SET UP PARAMETERS FOR VARIOUS ARGUMENT ITEMS
C
C   IF (TYPNAM .EQ. 'EULJOH ') THEN
C       function evaluation
C       R      = ARGVAL(1)
C       L      = ARGVAL(2)
C       E      = ARGVAL(3)
C       SIGMAC = ARGVAL(4)
C       SIGMA  = ARGVAL(5)
C       RGYRA  = R / 2.0
C       SLNDER = L / RGYRA

```



```

FACT = PI * SQRT(2.0D0 * E / SIGMAC)
FAC    = 1.0D0 - SIGMAC * (SLNDER) ** 2 / (4.0D0 * PI2 * E )
if ( forg .eq. 0 ) then
c      function evaluation
C      JOHNSON CRITERION
C      DR3VAL(1) = -SIGMA / (SIGMAC * FAC)
C      EULER CRITERION
C      DR3VAL(2) = -SIGMA * SLNDER**2 / (PI2 * E )
else if ( forg .eq. 1 ) then
c      gradient evaluation
do 10 iresp = 1, nresp
      do 20 iarg = 1, narg
          senval(iresp,iarg) = 0.0d0
      continue
10  continue
c      note that argval(2,3 and 4) are constant and therefore have
c      zero sensitivity
      dsldr = -slnder / r
      dfacr = -sigmac * slnder * dsldr / (2.0d0 *pi2 * e)
c      sensitivity of the first response to the first argument
      senval(1,1) = dfacr * sigma / ( sigmac * fac ** 2)
c      sensitivity of the second response to the first argument
      senval(2,1) = -2.0D0*sigma * slnder * dsldr /
1           (pi2 * e )
c      sensitivity of the first response to the second argument
      senval(1,2) = - 1.0d0 / (sigmac * fac)
c      sensitivity of the second response to the second argument
      senval(2,2) = -slnder**2 / (pi2 * e )
      else
          error = badfg
      endif
ELSE
    ERROR = BADTYP
END IF

RETURN
END

```

The subroutine gets the values of the response as a function of the five DRESP3 arguments. The user has to supply code for the computation of all DR3VAL(IR) and SENVAL(IR,IARG) for as many response types as defined in R3SGRT. In the sample routine taken above, the coding has been done only for TYPNAM = EULJOH. Note that only variable arguments are counted in IARG for SENVAL. In this case, IARG=1 is R which is ARGVAL(1) and IARG=2 is SIGMA which is ARGVAL(5). ARGVAL(2), ARGVAL(3) and ARGVAL(4) which are constants are not counted in IARG for SENVAL. The same subroutine is used for function evaluation and gradient evaluation. The FORG parameter specifies if this is a function evaluation (FORG=0) or a gradient evaluation (FORG=1).



Build External Servers using SCons Tool

Overview

MSC Nastran provides an external server capability (beam library, DRESP3 and Spline servers) to allow clients to create custom applications without modifying the Nastran program. Prior to MSC Nastran 2010, the binary server executables were built using make utilities. With the current release, they are built using SCons tool.

SCons is a Software Construction tool. Several benefits of using SCons tools are:

1. Automatically resolves source code dependencies.
2. Easy to build binary programs on different platforms with different operating systems. This is particularly true for building external server programs on Windows.
3. Easy to build new server programs when you add new source code. You can simply drop your source code to the target directory without the need to modify SConscript.
4. Easy to extend the current procedures to support applications that are written in Python scripts as well as in C or Fortran.

In this document, building DRESP3 servers is described. However, the discussion directly applies to the other two servers.

Data Structure Of External Servers

The location of a server directory is in the `install_dir/msc20121/nast/` directory. The supported beam library, dresp3 and spline servers are placed in three separate directories:

The root directory for beam library server: `install_dir/msc20121/nast/beamlib`

The root directory for DRESP3 server: `install_dir/msc20121/nast/dr3`

The root directory for Spline server: `install_dir/msc20121/nast/spline_server`

The installation directory for DRESP3 server is shown in the figure below. It is located in `install_dir/msc20121/nast/dr3` on UNIX and `install_dir\msc20121\nast\dr3` on Windows.



```

---- dr3 (root)
|
--- SConopts, SConscript, SConstruct
--- include
    |-- ftncalls.h, stdmsc.h, stdsystm.h
--- lib
    ----< ARCH 1>
        |
        ---- linux64
            | -- libdr3srv.a, cnxx.o, ..., semd.o
        ---- win64
            | -- dr3srv.lib, cnxx.obj, ..., semd.obj
    -----< ARCH i>
--- src
    |-- SConscript
    |-- dr3serv
        | -- SConscript, r3sgrt.F, r3svald.F, r3svals.F
    |
    | -- directories for other sample programs

```

Three SCons configuration files in the root directory are required by the SCons tool. Three subdirectories in the root directory are: `include`, `lib` and `src`.

- The `include/` directory contains general and operating system defines and defines macro to facilitate passing character strings between Fortran programs and C programs.
- The `lib/` directory includes predefined library and object files that are architecture dependent.
- The `src/` directory includes subdirectories that are created to hold source code for various sample server programs. For each server program, the Scons tool requires a subdirectory be created in the `src` directory. It is convenient to use the same program name for the directory name, but it is not required. For example, directory `dr3serv` is created for the `dr3serv` program. It contains a `SConscript` file and three Fortran template routines, `r3sgrt`, `r3svald`, `r3svals` described in Step 2: Write C or Fortran routines.

Build a DRESP3 Server

Building an external server requires your computer to have Software Development Kit installation (SDK) and the MSC Nastran installation with the external server option. (This is the standard installation and



should require no action on the user's part). For the purposes of this discussion, it is assumed the name of the program you are building is called dr3serv. In the dr3serv/ subdirectory, you have already modified three Fortran template routines. The binary program is saved as dr3serv on UNIX or dr3serv.exe on Windows.

The simplest way to build a server is to enter the command from the root directory,

```
scons dr3serv
```

where dr3serv is the name of the program you are building. By default, the command saves the program in the directory ~dr3/Apps/arch/bin that is architecture dependent. For example, arch=LX8664 indicates Linux 64 bit machine or arch=WIN8664 a Window 64 bit machine.

If you do not have the write privilege to the install_dir, you must copy the entire dr3/ directory to another location. Then enter the above command in the root directory to build the server.

The next several subsections present more advanced topics to customize your build, such as how to build a server from a subdirectory, how to redirect the output from a build to a new location and how to modify SConscript file for your special applications. For simple DRESP3 applications, this material can be skipped over for now and you can proceed to [Creating a CONNECT Command](#).

Build All the Programs in the ~src/ Directory

The current MSC supplied src/ directory contains 18 subdirectories or server programs. If you want the entire tree to be built, change to the root directory, (e.g., ~dr3/) and enter the command:

```
scons
```

This will build all the programs in the ~dr3/Apps/arch/bin directory.

Build Server Program(s) from a Subdirectory

If you want to build all the programs from a subdirectory, enter the following command in a subdirectory,

```
scons -D
```

If you are currently working in the dr3serv source directory and want to build only the dr3serv program from that directory, enter the following command

```
scons -D dr3serv
```

Store Server Program(s) in a Different Location

By default, the program is saved in the ~dr3/Apps/ directory. Specifying construction variables, APPS_LOCAL and SCA_OBJECT allows you to redirect the output from the build to another location. For example, you can use this option if you want to work from the system installation directory that you do not have the write access to and store the output in a subdirectory you do have write access to.

APPS_LOCAL is used to specify the location of Apps Local directory tree that contains the program(s) you have built and SCA_OBJECT is used to specify the location of the Object directory tree. To learn more about the SCons build environment, consult [Chapter 16 of SCA Framework User's Guide](#). These two variables may be specified in one of two ways:



1. Add an option to the scons command

```
scons dr3serv APPS_LOCAL=/scratch SCA_OBJECT=/scratch/.obj
or
```

2. Define construction variables APPS_LOCAL and SCA_OBJECT in the SConopts.user file. The SConopts.user is an optional SCons configuration file that is located in the home directory on your computer. The following shows a sample SConopts.user file:

```
# Sample of SConopts.user file
import sys
import SCASCons
#
# Object and Apps directory
if SCASCons.MACHINE == 'WINNT':
    SCA_OBJECT = 'C:/shz/msc2009t0/.obj'
    APPS_LOCAL = 'C:/shz/msc2009t0'
elif (SCASCons.MACHINE).startswith('LX'):
    SCA_OBJECT = '/scratch/shz/msc2009t0/.obj'
    APPS_LOCAL = '/scratch/shz/msc2009t0'
```

Build a Server Using Additional Source Files

Three Fortran routines, r3sgrt, r3svvald and r3svvals are template routines that provide a basis for your applications. You can extend them by simply arranging more lines in the template or calling a subroutine from the template. The subroutine may be written in Fortran and/or C. The SCons tool automatically processes all the source files in the directory that holds the SConscript file. So you simply drop the source codes in the directory without the need to modify the SConscript file.

After your code modification is complete for building a program called myserver1, you can enter the command from the subdirectory,

```
scons -D myserver1
```

Modify SConscript File to Build Custom Server Programs

If you want to provide a user-supplied name to your program or if you want to build the program by including your own library or object files, you need to modify the SConscript file. The SConscript file is written in Python scripts and is used to set any special construction variables unique to the directory that holds the file. It consists of three parts:

Standard and local library imports

This part is required to initialize the SCons construction environment and is placed at the top of the file.

```
import os
import time
Import("env_base")
env = env_base.Copy()
```

Standard ending scripts

They are exclusively reserved at the end of the file and should not be changed:



```

    retval = env.ProcessDir(env_base)
    Return('retval')

```

Local customizations.

Contains various construction variables that are used to

- a. specify different compiler options for different platforms,
- b. reference predefined global and machine dependent library and object files
- c. specify the name of the program to built

A sample of local customizations from the SConscript file is shown below. The highlighted lines will be further discussed in the following special applications.

```

# Start of local customization
#
# machine map ...
# Global and Local Compiling Options
#
env.Append(CPPDEFINES=['DRESP3_SERVER', '_MSCUTIL', '_OEM_NASTRAN'])
env.Append(MSCFPP_DEFINES = ['_IMPLICITNONE', 'DRESP3_SERVER'])
env.Append(CPPPATH=[src_top + os.sep + 'include'])
#
# ...
# Set up Local Variables
src_top=env.AbsDirPathInSou('#')

libdir=src_top + os.sep + 'lib' + os.sep + MscArch + os.sep

# Set Up Access to machine dependent Library and Object Files
if env['MACHINE'] == 'hpx':
    env.Append(LIBS=['nsl', 'pthread','c','dld'])
elif env['MACHINE'] == 'hpxipf':
    env.Append(LIBS=['nsl'])

elif env['MACHINE'] == 'solaris':
    env.Append(LIBS=['aio','rt','sunmath','socket','nsl','intl'])
    env.Append(LIBS=['Cstd','Crun','malloc','dl','mvec'])

elif env['MACHINE'].startswith('WIN'):
    env.AddLinkLibrary(src_top + '../../../../../api/' + MscArch + '/xdr')
    env.Append(LIBS=['netapi32', 'Advapi32', 'ws2_32'])
    env.AddLinkSource(libdir+'getlserm'+env['OBJSUFFIX'])

# Set Up Access to Global Library and Object Files
env.AddLinkSource(libdir+'cnxx'+env['OBJSUFFIX'])
env.AddLinkSource(libdir+'initgmsrvcmns'+env['OBJSUFFIX'])
env.AddLinkSource(libdir+'main'+env['OBJSUFFIX'])
env.AddLinkSource(libdir+'semd'+env['OBJSUFFIX'])
env.AddLinkLibrary(libdir+'dr3srv')

# Specify Name of the Program to be Built and Other Options.
env.BuildProgram('dr3serv', aliases=None, tree='apps', fortranmain=False,
fortranlibs=True)

# End of local customization

```

Now we are ready to modify the SConscripts file and build servers with special requirements.

1. Build a Server with a User-Supplied Name

If you want to create a program with a selected name, follow the steps below and assume the server is called mydr3serv.



- a. Create a new subdirectory called mydr3serv (or any other name) in the src/ directory.
- b. Change to the new directory and create files SConscript, r3sgrt, r3svld, r3svls by copying from an existing directory, say dr3serv/.
- c. Edit SConscript and locate env.BuildProgram line as highlighted above. The program name given in the env.BuildProgram() must match that of the program you are building.


```
env.BuildProgram('dr3serv', aliases=None, tree='apps', ...) # before
env.BuildProgram('mydr3serv', aliases=None, tree='apps', ...) # after
```
- d. Modify the Fortran routines as needed

Enter the following command from your working directory,

```
scons -D mydr3serv
```

Or change directory to the root and enter command

```
scons mydr3serv
```

If the command runs successfully, the binary program, mydr3serv should be created.

2. Build a Server Using Additional Library and Object Files

You can build a program while including your own library or object files. Assume your server needs one custom library called libmydr3.a and two object files called mydr3obj1.o and mydr3obj2.o on UNIX or mydr3.lib, mydr3obj1.obj and mydr3obj2.obj on Windows. Further assume your new program is called myserver2. Follow the steps below to complete this special task.

- a. Change to the lib/ directory and place your three library and object files there.
- b. Create a new directory called myserver2 under ~src/ and change to that directory.
- c. Copy SConscript file and three Fortran routines from an existing sample directory.
- d. Edit SConscript file,

Locate env.AddLinkLibrary as highlighted in the sample local customization shown above. The existing python script references the existing library, dr3srv stored in ~lib directory. Notice local variable libdir has been defined for the ~dr3/lib/ directory. It also references another local variable, src_top that defines the path of the root directory. Therefore, inserting the following script will allow the SCons tool to including your own library, mydr3 in your server build:

```
Env.AddLinkLibrary(libdir+'mydr3')
```

- e. Locate env.AddSource as highlighted above. This python script line is used for each object file in the ~lib directory. In order to include two of your own object files, you need to insert the following lines:

```
env.AddLinkSource(libdir+'mydr3obj1'+env['OBJSUFFIX'])
env.AddLinkSource(libdir+'mydr3obj2'+env['OBJSUFFIX'])
```

Notice variable env['OBJSUFFIX'] automatically resolves architecture dependent file extension for an object file: the form of .o on UNIX and the form of .obj on Windows.



f. To build the server from the subdirectory you are working on, enter the following command

```
scons -D myserver2
```

3. Special procedure for Window Platforms

If you are building a server program on Windows in a non-system installation directory, you must take the following extra steps to avoid reference to a missing `xdr` library.

a. First locate the `xdr` library file. Assume it has been located at

`d:\bld\nastran\msc2012\api\win64\xdr.lib`. Open the `SConscript` file in the target subdirectory.

b. Define local variable, `libdir1` to reference the path holding the `xdr` library: `libdir1='d:' + os.sep + 'bld' + os.sep + 'nastran' + os.sep + 'msc2012' + os.sep + 'api' + os.sep + MscArch + os.sep`

c. Locate an existing script line,

```
env.AddLinkLibrary(src_top + '/../../api/' + MscArch + '/xdr')
```

and replace it with the new script line:

```
env.AddLinkLibrary(libdir1 + 'xdr').
```

Useful SCons Build Options

There are many options you may specify on a SCons command. Two options described in this documents are: `Buildtype` and `Debugprint`. To learn more, consult Chapter 16 of SCA Framework User's Guide.

Buildtype

specifies a compiler option for the build. If you like to run a debugger program (assuming it is available from your system) to detect possible coding errors, you need to build your program with a debug compile. This can be achieved by entering the command

```
scons dr3serv debug=yes
```

Debugprint

This option requests the SCons tool to print out build diagnostic messages by adding `debugprint=i` ($i=0$ to 3) to the command. The following command with `debugprint=3` will print out the full set of diagnostic messages such as source trees scanning, compiling and linking information of each subroutine used in your server program.

```
scons dr3serv debugprint=3
```

Creating a CONNECT Command

The CONNECT command, specified in the File Management System (FMS) section, defines the response group. Multiple CONNECT entries may be used to define multiple groups. This command is discussed in File Management.

For continuity, the format is repeated here:



CONNECT DRESP3 Group_name Evaluator_name

Example

CONNECT DRESP3 MYBUCK EXTRESP

Creating an Evaluator Connection File

The evaluator connection file defines the association between the response group and the corresponding server program.

Format:

Evaluator_name, the connection option, the path of the server program

Example for evaluator connection file extconnect:

EXTRESP,-,/net/harkness/users/shz/dr3srv/dr3serv on UNIX

or

EXTRESP,-,D:\Scratch\DRSP3\dr3serv.exe on Windows

where EXTRESP is the evaluator name, symbol '-' indicates that the pipe option is used for the association (i.e., server executable program resides in the same computer as the Nastran program does). The last term is the path name where the server program, dr3serv resides.

Submitting a Nastran Job with the gmconn Keyword

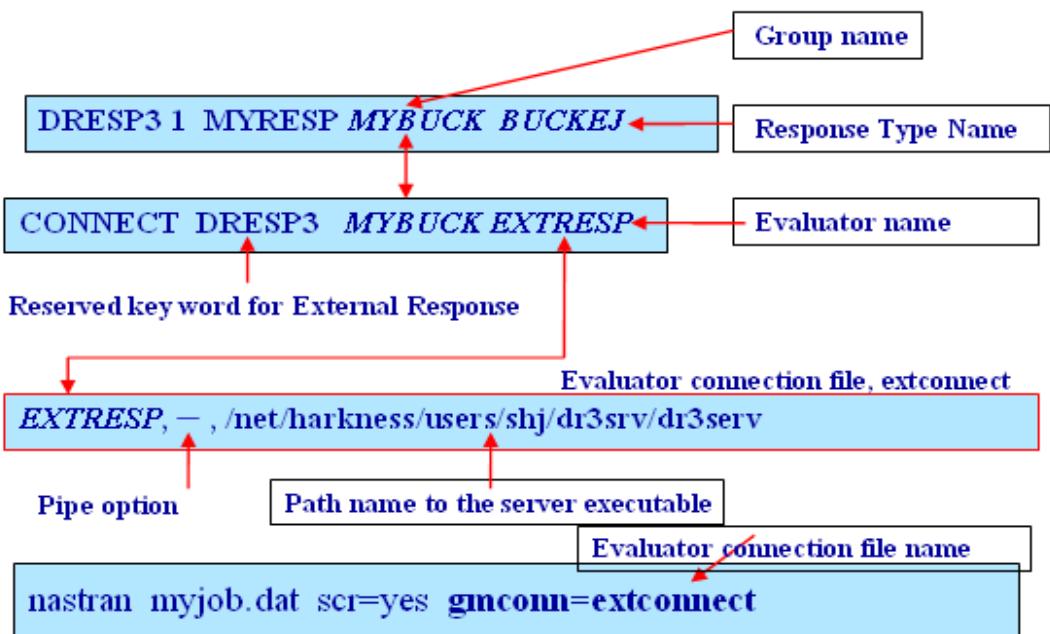
gmconn is the keyword used on the command line to reference the name of the evaluator connection files. The connection file is processed at the Nastran initialization stage to establish a physical link between the Nastran program and the server program(s).

Example:

mscnast2012 myjob scr=yes gmconn=extconnect



Summary of User Interface



Guidelines and Limitations

It is important to realize that the gradients that are provided are for the responses with respect to the DRESP3 arguments and not (necessarily) the design variables. This takes the burden of performing the chain rule calculations from the user and uses existing Nastran operations to compute terms such as:

$$\frac{dr_3}{dx} = \frac{\delta r_3}{\delta x} + \sum \frac{\delta r_3}{\delta r_i} \frac{\delta r_i}{\delta x}$$

Instead, the R3SVALD subroutine provides the $\delta r_3/\delta x$ and $\delta r_3/\delta r_i$ terms and the remaining operations are performed within Nastran.

Limitation

There is a current limitation that all GRDTYP's for a particular TYPE must be the same. The GRDTYP's do not need to all be the same for all the DRESP3's in an input file. That is, one can specify analytic gradients for one TYPE and finite difference gradients for another type.

Useful Feature for the Case of a Large Number of DRESP1 Entries and Few Variables

DRESP3's which have a large number of DRESP1's and no other types of data have been given a special gradient path. It makes sense to do the finite difference gradient calculation by perturbing all the DRESP1 quantities for a particular design variable and then calling DRESP3 evaluation. In this way, the number of calls to the evaluator is reduced from $2*NRSP1$ to $2*NDVI$. When $NRSP1 \gg NDVI$, this can provide a major performance improvement.





5

Output Features and Interpretation

- Output-Controlling Parameters 267
- Design Optimization Output 269
- Design Sensitivity Output 291
- Design Punch Output 297
- Postprocessing Output 303



A MSC Nastran run will produce a number of output files and the goal of this chapter is to explain the features of these files that are unique to Design Sensitivity and Optimization. Results shown in the .f06 file are emphasized, but attention is also given to results in the .pch (punch) file, a special “comma separated values” file and in two possible output database files: .op2 and .xdb.

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. There are a few parameters that affect the level of detail and the frequency of output (for design cycle-dependent data). Although covered elsewhere in this guide, these parameters are summarized in this chapter in [Output-Controlling Parameters](#).

This is followed by [Design Optimization Output](#) that serves to document the printed output that is unique to design optimization through the use of the output from a design optimization problem (DSOUG4.DAT) in the Test Problem Library (TPL) supplied with your MSC Nastran installation tapes. This example introduces most of the frequently encountered types of design optimization output, within the context of an actual problem. The section also documents prints that occur for some of the specialized features of design optimization, such as Fully Stressed Design. The [Design Sensitivity Output](#) is a similar output example for design sensitivity analysis. Punch output is described in [Design Punch Output](#) while several postprocessing options are described in [Postprocessing Output](#).



Output-Controlling Parameters

Parameters that affect either the frequency or level of detail of the output are discussed here briefly. Some of them (namely, IPRINT, P1, P2, and P2xxxx) are specified on the DOPTPRM Bulk Data entry described in [Bulk Data Entries](#) while the remainder are input using the PARAM Bulk Data entry. More information on many of these same parameters can be obtained from [Parameters Unique to Design Sensitivity and Optimization](#).

DESPCH	Controls the frequency updated Bulk Data entry output to the punch file. The default is for the final design cycle only, but can be changed to every n-th design cycle (plus the last) with this parameter.
DESPCH1	Controls the amount and form of updated Bulk Data entry output to the punch file. The amount refers to updated properties, DESVAR, DRESP1 and GRID entries. The sign selects between short and large field formats.
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 Bulk Data entry, DOPTPRM .
NASPRT	Governs the frequency of MSC 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.
OPTEXIT	When set to a value of 4 or 7, provides output of the unformatted 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. This parameter has been superseded by the DSAPRT Case Control command.
P1	Controls the frequency of design cycle output. By default, initial and final optimization summaries are output but can be changed to every n-th cycle using this parameter.
P2	Controls how much design cycle information is output at the intervals determined by P1.
P2CR	Maximum number of Constraints on Responses to be printed.
P2CDDV	Maximum number of Constraints on Dependent Design Variables to be printed.
P2CP	Maximum number of Constraints on Properties to be printed.
P2CC	Maximum number of Constraints on Connectivity properties to be printed.
P2CM	Maximum number of Constraints on Material properties to be printed.
P2CBL	Maximum number of Constraints on Beam Library dimensions to be printed.
P2CALL	Maximum number of Constraints of all categories to be printed.



P2RSET	SET1 ID for which a set of Response IDs is defined.
POST	This Bulk Data parameter is used to identify the data base (.xdb or .op2) to be used for storing pre- and postprocessing data and, in the case of the .op2 file, what postprocessor is to be used. In the context of design optimizaton, PARAM POST-1 is the most useful in that it creates data blocks that can be used to perform xy plotting in Patran as shown in Postprocessing Output .



Design Optimization Output

This section presents design sensitivity and optimization output in the context of one of the example problems (DSOUG4) from MSC Nastran's test problem library that is also presented as the [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 the reader to locate this problem in the test problem library and run it on his/her installation. The output from the actual problem can then be referenced while reading this section. The results shown here were generated using MSC Nastran 2014.1 on an Intel-based PC running Linux. Your output should be quite similar, but there may be slight differences.

DSOUG4 Output

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 TYPE	PROPERTY ID	PROPERTY NAME	ANALYSIS VALUE	DESIGN VALUE	LOWER BOUND	UPPER BOUND	DIFFERENCE FLAG	SPAWNING FLAG
PBAR	3	A	9.600000E-01	9.600000E-01	1.000000E-15	1.000000E+20	NONE	*
PBAR	3	I1	1.207200E+00	1.207200E+00	1.000000E-15	1.000000E+20	NONE	*
PBAR	3	I2	9.812000E-01	9.812000E-01	1.000000E-15	1.000000E+20	NONE	*
PBAR	3	J	3.223445E-03	3.223445E-03	1.000000E-15	1.000000E+20	NONE	*
PBAR	3	C1	1.500000E+00	1.500000E+00	N/A	N/A	NONE	*
PBAR	3	D1	-1.500000E+00	-1.500000E+00	N/A	N/A	NONE	*
PBAR	3	E1	-1.500000E+00	-1.500000E+00	N/A	N/A	NONE	*
PBAR	3	F1	1.500000E+00	1.500000E+00	N/A	N/A	NONE	*
PBAR	3	K1	5.477937E-01	5.477937E-01	N/A	N/A	NONE	*
PBAR	3	K2	1.200344E-01	1.200344E-01	N/A	N/A	NONE	*
PBARL	3	DIM2	1.000000E-01	1.000000E-01	N/A	N/A	NONE	*
PSHELL	1	T	1.500000E-01	1.500000E-01	1.000000E-02	1.000000E+20	NONE	

1. 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 AND THE DESIGN MODEL IS BEING USED TO OVERRIDE THE ANALYSIS MODEL.
IF THE FLAG IS FATAL, THEN THE DIFFERENCES ARE GREATER THAN 1.00000E+35 AND THE RUN WILL BE TERMINATED.
2. THE SPAWNING FLAG (*) INDICATES THAT THE SPAWNED PROPERTY IS DERIVED EITHER FROM THE BEAM CROSS SECTION LIBRARY OR FROM A PBEAM ENTRY. THE PROPERTY ID FOR THE SPAWNED PROPERTY IS IDENTICAL TO ITS PARENT.

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

ELEMENT TYPE	ELEMENT ID	CONNECTIVITY NAME	ANALYSIS VALUE	DESIGN VALUE	LOWER BOUND	UPPER BOUND	DIFFERENCE FLAG
CBAR	31	W3A	1.575000E+00	1.575000E+00	N/A	N/A	NONE
CBAR	31	W3B	1.575000E+00	1.575000E+00	N/A	N/A	NONE
CBAR	32	W3A	1.575000E+00	1.575000E+00	N/A	N/A	NONE
CBAR	32	W3B	1.575000E+00	1.575000E+00	N/A	N/A	NONE
CBAR	33	W3A	1.575000E+00	1.575000E+00	N/A	N/A	NONE
CBAR	33	W3B	1.575000E+00	1.575000E+00	N/A	N/A	NONE
CBAR	34	W3A	1.575000E+00	1.575000E+00	N/A	N/A	NONE
CBAR	34	W3B	1.575000E+00	1.575000E+00	N/A	N/A	NONE

1. 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 AND THE DESIGN MODEL IS BEING USED TO OVERRIDE THE ANALYSIS MODEL.

IF THE FLAG IS FATAL, THEN THE DIFFERENCES ARE GREATER THAN 1.00000E+35 AND THE RUN WILL BE TERMINATED.

Note that this example has two sets of properties that are being compared: the “properties” that are defined on a property Bulk Data entry and designed using a DVPRELi Bulk Data entry and the “connectivity properties that are defined on connectivity bulk data entries (those that begin with the letter ‘c’) and that are designed using the DVCRELI Bulk Data entries. If there had been designed material properties in the example, this would have caused the print of an additional table that is a



“COMPARSION BETWEEN INPUT MATERIAL PROPERTY VALUES FROM ANALYSIS AND DESIGN MODELS.” This comparison data is only printed once, at the beginning of the design task and it cannot be “turned off.”

The point of these tables is to allow you to check if the design values in your model are what you expect and to see if they differ from the analysis values. If any of the analysis model properties differ 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 for any of the properties defined on property entries or on connectivity properties. Note that the first table lists a series of PBAR properties that have not been explicitly designed in the input file. Instead, the second beam dimension on the PBARL entry (which corresponds to the thickness of a HAT beam cross section) is designed and this causes 10 PBAR properties to be designed, including four stress recovery points. These derived or “spawned” properties are indicated by an asterisk in the final field of the table.

The upper and lower bounds that are printed in these tables correspond to the PMIN and PMAX values provided on the DVPREL1 and DVCREL1 entries. The “spawned” properties that are a nonlinear function of the design variable have default values applied to these bounds. The DIM2 property of the PBARL, the “spawned” properties that are a linear function of the design variable and all of the material properties have bounds of “N/A” (not applicable) indicating that no bounds are imposed on these properties that are linear functions of a single design variable.

If the DIFFERENCE FLAG has a FATAL value, this indicates that the analysis model and design model differ by such a large degree that the design task will be stopped. This should not happen in normal circumstances, but the discussion of the PTOL parameter in the [Comparison Between the Design and Analysis Properties \(PTOL, PLVOL\)](#) portion of the DOPTPRM description shows how the allowable difference between the analysis and designed property values can be made arbitrarily small.

Once the analysis and design models have been compared, 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:

```
^^^ USER INFORMATION MESSAGE 9051 (FEA)
^^^ STATIC ANALYSIS INITIATED. DESIGN CYCLE NUMBER=
```

1



Since this is the first analysis and the parameter NASPRT is 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 MSC Nastran static analysis output (abbreviated here):

LOAD CONDITION 1							SUBCASE 1		
DISPLACEMENT VECTOR									
POINT ID.	TYPE	T1	T2	T3	R1	R2	R3		
10000	G	0.0	0.0	0.0	-1.951458E-02	1.643483E-03	0.0		
10001	G	1.310807E-03	-7.580876E-05	0.0	2.822207E-02	-7.535636E-05	0.0		
10002	G	2.869767E-03	-2.416459E-04	0.0	3.423017E-02	4.365287E-06	0.0		
10003	G	5.090274E-03	-6.282286E-05	0.0	2.823293E-02	7.113095E-05	0.0		
10004	G	8.191596E-03	1.850179E-03	0.0	-1.949971E-02	-1.643704E-03	0.0		
10100	G	0.0	-3.821074E-04	0.0	1.326969E-02	-1.979090E-02	0.0		
..							
..							
..							
..							

LOAD CONDITION 2							SUBCASE 2		
ELEMENT ID.	FIBER DISTANCE	STRESSES IN ELEMENT COORD SYSTEM			ELEMENTS (QUAD4)			MINOR	
		NORMAL-X	NORMAL-Y	SHEAR-XY	PRINCIPAL STRESSES (ZERO SHEAR)	ANGLE	MAJOR		
VON MISES									
1	-7.500000E-02	-1.741460E+03	3.724309E+01	2.401960E+02	82.4431	6.910829E+01	-1.773325E+03		
1.808870E+03	7.500000E-02	-1.847658E+03	2.857752E+02	-5.812001E+02	-75.7081	4.338334E+02	-1.995717E+03		
2.244305E+03	-7.500000E-02	-1.766428E+03	-1.469386E+02	-1.724371E+02	-83.9892	-1.287818E+02	-1.784585E+03		
1.723806E+03	7.500000E-02	-1.055006E+03	4.101843E+02	-8.230811E+02	-65.8356	7.794776E+02	-1.424299E+03		
1.935568E+03	3	-7.500000E-02	-5.709843E+02	-4.379282E+02	-3.948668E+02	-49.7818	-1.040243E+02	-9.048882E+02	
8.576207E+02	7.500000E-02	-6.261518E+02	-8.278744E+01	-6.108423E+02	-56.9889	3.140658E+02	-1.023005E+03		
1.210978E+03	4	-7.500000E-02	5.041106E+02	-1.189463E+02	-6.960353E+02	-32.9439	9.551535E+02	-5.699892E+02	
1.334778E+03	7.500000E-02	-3.647220E+02	5.339794E+01	9.883966E+02	50.9714	8.546022E+02	-1.165926E+03		
1.756739E+03									

This is followed by the DMAP information message:

```
^^^ USER 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 performs a hard convergence check when this is 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 : -1.6392E-01
-----
```

Recall that these constraint values refer to the normalized constraints constructed internally in MSC Nastran. Since the maximum constraint value is negative, we can characterize this initial design as feasible. Furthermore, its value of -0.16392 indicates a constraint satisfaction of around 16%. We will see shortly how to determine which constraint is responsible for this maximum value.

The next set of design related prints in the output is controlled by the P1 and P2 parameters just discussed in [Output-Controlling Parameters](#). The example has set P1=1 and P2=15 so that all available design data are printed. An abbreviated version of these prints is presented here interspersed in the text. In this case, the prints follow the initial analysis so there is no redesign information available to print. Instead, initial data, as indicated by the following prints, is presented.



```
*****
*
*
*          D E S I G N      O P T I M I Z A T I O N
*
*****
*****
```



```
*****
*
*          I N I T I A L      A N A L Y S I S
*
*****
*****
```



```
*****      ANALYSIS RESULTS BASED ON THE INITIAL DESIGN *****
```



```
----- DESIGN OBJECTIVE -----
```

INTERNAL RESPONSE ID	DRESPx	RESPONSE TYPE	MINIMIZE OR MAXIMIZE	SUPERELEMENT ID	SUBCASE ID	VALUE
1	DRESP1	WEIGHT	MINIMIZE	0	0	6.9618E+00


```
----- DESIGN VARIABLES -----
```

INTERNAL ID	DESVAR ID	LABEL	LOWER BOUND	VALUE	UPPER BOUND
1	1	T-PLATE	1.0000E-03	1.5000E-01	1.0000E+01
2	2	HATDIM2	1.0000E-03	1.0000E+00	1.0000E+01

The DESIGN OBJECTIVE print provides attributes of the objective and its initial value. This is followed by a table DESIGN VARIABLES that contains information included on the DESVAR, including the initial values of the design variables. Two IDs are provided each design variable: an internal ID that indicates the position of the design variable in the design task and the user provided DESVAR ID. Dependent design variables that are defined using DLINK entries always follow the independent design variables. The identity of these design variables is useful when attempting to analyze the output from the optimizer algorithm and when interpreting unformatted design sensitivity prints.

The designed properties are presented next and it is again seen that a distinction is made in the type of property in that standard properties are presented in a table separate from connectivity properties (and material properties if any are present in the design). Also spawned beam properties are designated as having a SECPRO type of property. The initial value of the property is bracketed by its lower and upper bounds. Again, properties that are a linear function of a single design variable and have no user designed limits do not have bounds.



DESIGNED PROPERTIES						
PROPERTY TYPE	PROPERTY ID	PROPERTY NAME	TYPE OF PROPERTY	LOWER BOUND	VALUE	UPPER BOUND
PSHELL	1	T	DVPREL1	1.0000E-02	1.5000E-01	1.0000E+20
PBARL	3	DIM2	DVPREL1	N/A	1.0000E-01	N/A
PBAR	3	A	SECPRO	1.0000E-15	9.6000E-01	1.0000E+20
PBAR	3	I1	SECPRO	1.0000E-15	1.2072E+00	1.0000E+20
PBAR	3	I2	SECPRO	1.0000E-15	9.8120E-01	1.0000E+20
PBAR	3	J	SECPRO	1.0000E-15	3.2000E-03	1.0000E+20
PBAR	3	C1	SECPRO	-1.0000E+35	1.5000E+00	1.0000E+20
PBAR	3	D1	SECPRO	-1.0000E+35	-1.5000E+00	1.0000E+20
PBAR	3	E1	SECPRO	-1.0000E+35	-1.5000E+00	1.0000E+20
PBAR	3	F1	SECPRO	-1.0000E+35	1.5000E+00	1.0000E+20
PBAR	3	K1	SECPRO	-1.0000E-03	5.8333E-01	1.0000E+20
PBAR	3	K2	SECPRO	-1.0000E-03	3.7500E-01	1.0000E+20

DESIGNED CONNECTIVITY PROPERTIES						
ELEMENT TYPE	ELEMENT ID	PROPERTY NAME	TYPE OF PROPERTY	LOWER BOUND	VALUE	UPPER BOUND
CBAR	31	W3A	DVCREL1	N/A	1.5750E+00	N/A
CBAR	32	W3A	DVCREL1	N/A	1.5750E+00	N/A
CBAR	33	W3A	DVCREL1	N/A	1.5750E+00	N/A
CBAR	34	W3A	DVCREL1	N/A	1.5750E+00	N/A
CBAR	31	W3B	DVCREL1	N/A	1.5750E+00	N/A
CBAR	32	W3B	DVCREL1	N/A	1.5750E+00	N/A
CBAR	33	W3B	DVCREL1	N/A	1.5750E+00	N/A
CBAR	34	W3B	DVCREL1	N/A	1.5750E+00	N/A

The property prints are followed by constraint prints and there can be up to four types of constraints printed. The first table shows the constraints on the design responses. These are the constraints that are imposed on DRESP1/2/3 responses. The constraints each have an internal ID and this is useful in analyzing the detailed optimization prints discussed on [page 267](#). The internal constraint ID is followed by a DCONSTR ID that was used in the definition of the constraint and then has the internal response ID. This internal ID can be used to identify the response that resulted in the constraint, as will be seen in a moment when we discuss how to trace the identity of the response that results in the maximum constraint. The response type (RTYPE on the DRESP1 entry) is then given, as is a flag indicating whether the constraint arose from a lower bound or upper bound limit. The region ID used in the screening process is followed by the subcase ID and finally the constraint value.

DESIGN CONSTRAINTS ON RESPONSES								
(MAXIMUM RESPONSE CONSTRAINTS MARKED WITH **)								
INTERNAL ID	DCONSTR ID	INTERNAL RESPONSE ID	EXTERNAL DRESPX ID	RESPONSE TYPE	L/U FLAG	INTERNAL REGION ID	SUBCASE ID	VALUE
1	10	2	3	STRESS	UPPER	2	1	-4.3148E-01
2	10	3	3	STRESS	UPPER	2	1	-3.9778E-01
3	10	4	3	STRESS	UPPER	2	1	-4.3148E-01
4	10	5	3	STRESS	UPPER	2	1	-3.9778E-01

DESIGN CONSTRAINTS ON RESPONSES								
(MAXIMUM RESPONSE CONSTRAINTS MARKED WITH **)								
INTERNAL ID	DCONSTR ID	INTERNAL RESPONSE ID	EXTERNAL DRESPX ID	RESPONSE TYPE	L/U FLAG	INTERNAL REGION ID	SUBCASE ID	VALUE
5	10	6	6	STRESS	UPPER	2	1	-4.6689E-01
6	10	7	6	STRESS	UPPER	2	1	-4.5410E-01
7	10	8	6	STRESS	UPPER	2	1	-4.6689E-01
8	10	9	6	STRESS	UPPER	2	1	-4.5410E-01
9	10	11	1	STRESS	UPPER	1	2	-1.6392E-01**
10	10	12	2	STRESS	UPPER	1	2	-4.9124E-01
11	10	13	2	STRESS	UPPER	1	2	-2.2041E-01



12	30	10	14	DISP	UPPER	14	2	-4.9579E-01
----- CONSTRAINTS ON DESIGNED PROPERTIES -----								
INTERNAL ID	PROPERTY ID	PROPERTY NAME	L/U FLAG	CYCLE LIMIT	VALUE			
13	1	T	LOWER	7.5000E-02	-1.0000E+00			
14	3	A	LOWER	4.8000E-01	-1.0000E+00			
15	3	I1	LOWER	6.0360E-01	-1.0000E+00			
16	3	I2	LOWER	4.9060E-01	-1.0000E+00			
17	3	J	LOWER	1.0000E-15	-3.2234E+12			
18	1	T	UPPER	2.2500E-01	-3.3333E-01			
19	3	A	UPPER	1.4400E+00	-3.3333E-01			
20	3	I1	UPPER	1.8108E+00	-3.3333E-01			
21	3	I2	UPPER	1.4718E+00	-3.3333E-01			
22	3	J	UPPER	1.3223E-02	-7.5623E-01			
----- CONSTRAINTS ON DESIGNED BEAM LIBRARY DIMENSIONS -----								
INTERNAL ID	PBART ID	GROUP	TYPE	CONSTRAINT NO.	VALUE			
23	3	MSCBML0	HAT	1	-2.8000E+00			
24	3	MSCBML0	HAT	2	-1.8000E+00			

The header for the DESIGN CONSTRAINT ON RESPONSES also includes the notation: “MAXIMUM RESPONSE CONSTRAINTS ARE MARKED WITH **”. In this case, we see that the ninth internal constraint has a value of -0.16392 (the value given from the hard convergence check above) and that the corresponding internal response ID of this maximum constraint is 11. To see which response this is, we skip ahead to the table on the next page. It is seen that the eleventh internal response is a stress response in a CBAR element 34 and that the stress occurs in the second subcase and is component 7 (the maximum stress at End A of the element). It is also seen that the satisfied constraint results from the stress being less than the allowable value of 25,000.

Returning to the constraint printout, the next type of constraint that could be printed is those resulting from constraints placed on dependent design variables. In this example, there are no dependent design variables, so there is no print for this type of constraint. There are constraints applied to the properties however and these occur next in the printout. The internal constraint numbering continues from the response constraints and the table then lists the property ID and name, an indication as to whether the constraint comes from a lower or upper bound, the cycle limit and then the constraint value. The cycle limit is the limit placed on the property for the current approximate model and, as explained in the [Move Limits](#), this limit is computed using the current value of the property, and the user applied limits and the DELP and DPMIN parameters. A comparison of the list of constrained properties in this printout with the designed properties of the previous fragment indicates that not all designed properties are constrained. As the [Move Limits](#) has indicated, a property is not constrained when

1. it is a linear function of a single design variable, and
2. the user has not imposed any limits on the property.

These two criteria are satisfied for the DIM2 PBART dimension, the BAR stress recovery points and for all the connectivity properties. The thickness of the PSHELL is a linear function of a single design variable, but since it has been limited to be greater than 0.01, there is a move limit constraint.

The final constraint type is associated with the beam library. It is seen that there are two of these constraints and that they are applied to the PBART with PID=3 and that the section type is a HAT. Referring to [Table 2-6](#), it is seen that the two constraints are to insure that twice the thickness (DIM2) never exceeds the height (DIM1) of the HAT or the center (DIM3) of the HAT. In this case, the initial



thickness is 0.1, the (undesigned) height is 3.0 and the width is 2.0, resulting in constraint values of -2.8 and -1.8, respectively.

Note that the maximum constraint that is marked with the ** is only for the constraints that derive from the responses. In this case, this happens to be the maximum response across all response types, but it is possible to have situations where the maximum constraint comes from one of the other constraint types and it will not be flagged.

The final P1/P2 associated prints are associated with the responses themselves. Each response type has specialized formats to provide the information and it would be tedious to describe each of the possible formats. Each of the responses does have an internal ID and we have already seen how this can be used to link the constraints on the responses to the responses themselves. The DRESP1 responses are sorted by superelement, by subcase and by response type. If there are DRESP2 responses, they follow the DRESP1 responses for a given superelement/subcase and DRESP3 responses follow the DRESP2s.

RESPONSES IN DESIGN MODEL								
(N/A = BOUND NOT ACTIVE OR AVAILABLE) (** VIOLATED RESPONSES MARKED WITH V ***) (** ACTIVE RESPONSES MARKED WITH A ***)								
----- WEIGHT RESPONSE -----								
INTERNAL ID	DRESP1 ID	RESPONSE LABEL	ROW ID	COLUMN ID	LOWER BOUND	VALUE	UPPER BOUND	
1	15	W	3	3	N/A	6.9618E+00	N/A	
INITIAL ANALYSIS SUBCASE = 1								
----- STRESS RESPONSES -----								
INTERNAL ID	DRESP1 ID	RESPONSE LABEL	ELEMENT ID	VIEW ELM ID	COMPONENT NO.	LOWER BOUND	VALUE	UPPER BOUND
2	3	S13	2		9	N/A	1.4213E+04	2.5000E+04
3	3	S13	3		9	N/A	1.5056E+04	2.5000E+04
4	3	S13	14		9	N/A	1.4213E+04	2.5000E+04
5	3	S13	15		9	N/A	1.5056E+04	2.5000E+04
6	6	S16	2		17	N/A	1.3328E+04	2.5000E+04
7	6	S16	3		17	N/A	1.3648E+04	2.5000E+04
8	6	S16	14		17	N/A	1.3328E+04	2.5000E+04
9	6	S16	15		17	N/A	1.3648E+04	2.5000E+04
INITIAL ANALYSIS SUBCASE = 2								
----- DISPLACEMENT RESPONSES -----								
INTERNAL ID	DRESP1 ID	RESPONSE LABEL	GRID ID	COMPONENT NO.	LOWER BOUND	VALUE	UPPER BOUND	
10	14	D2	10203	3	N/A	1.5126E-02	3.0000E-02	
----- STRESS RESPONSES -----								
INTERNAL ID	DRESP1 ID	RESPONSE LABEL	ELEMENT ID	VIEW ELM ID	COMPONENT NO.	LOWER BOUND	VALUE	UPPER BOUND
11	1	SBARA	34		7	N/A	2.0902E+04	2.5000E+04
INITIAL ANALYSIS SUBCASE = 2								
----- STRESS RESPONSES -----								
INTERNAL ID	DRESP1 ID	RESPONSE LABEL	ELEMENT ID	VIEW ELM ID	COMPONENT NO.	LOWER BOUND	VALUE	UPPER BOUND
12	2	SBARB	32		14	N/A	1.2719E+04	2.5000E+04
13	2	SEARB	33		14	N/A	1.9490E+04	2.5000E+04



After the initial design information has been printed, two messages appear in the output to indicate that a sensitivity analysis has been performed.

```
^^^ USER INFORMATION MESSAGE 9051 (FEA)
^^^ STATIC SENSITIVITY ANALYSIS INITIATED. DESIGN CYCLE NUMBER= 1
^^^ USER INFORMATION MESSAGE 9052 (FEA)
^^^ STATIC SENSITIVITY ANALYSIS COMPLETED. DESIGN CYCLE NUMBER= 1
```

Unless the DSAPRT Case Control command has been used to request sensitivities for this design cycle, no prints are produced during the sensitivity calculations. Sensitivity prints are discussed in the next section: [Design Sensitivity Output](#). After the sensitivity analysis has been performed, all of the information necessary to perform an optimization is available.

The DSOUG4 deck as present in the tpl also produces a large amount of output based on the DOPTPRM parameter IPRINT. IPRINT = 7 in this example, the most verbose value, produces information on the optimization parameter and the initial design in terms of the objective, design variables and constraints as seen by the optimizer. It can also produce gradient information, search direction information and then results from the one-dimensional search. After the optimization for the current cycle is complete, information on the final objective, design variables and constraints are printed as is information on the time spent in different phases of the optimization task. Typically, this is more information than is needed for a SOL 200 analysis and would be invoked only if, for example, the approximate optimization tasks fails or leads to a design that does not appear credible. This IPRINT=7 information is not presented here.

After the approximate optimization summary, output for the first design cycle, produced by the P1 = 1, P2 = 15 parameters on the DOPTPRM entry, is presented. This is the same type of output as we discussed starting on [page 272](#) except that now we have data available from the input and the output of the approximate optimization problem. The input values are identical to the initial values for the design and the tables of data have the same number of rows so only an abbreviated set of these tables are shown here. This summary begins with the identification of the design cycle number and this is followed by 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
*
*****
*****          OPTIMIZATION RESULTS BASED ON THE APPROXIMATE MODEL      *****
----- DESIGN OBJECTIVE -----
----- INTERNAL RESPONSE ID      DRESPx      RESPONSE TYPE      MINIMIZE OR MAXIMIZE      SUPERELEMENT ID      SUBCASE ID      INPUT VALUE      OUTPUT VALUE -----
1          DRESP1          WEIGHT          MINIMIZE                  0                  0      6.9618E+00      5.3694E+00
----- DESIGN VARIABLES -----

```



INTERNAL ID	DESVAR ID	LABEL	LOWER BOUND	INPUT VALUE	OUTPUT VALUE	UPPER BOUND
1	1	T-PLATE	1.0000E-03	1.5000E-01	1.0851E-01	1.0000E+01
2	2	HATDIM2	1.0000E-03	1.0000E+00	8.4327E-01	1.0000E+01

Note that the first design variable has decreased while the second has increased.

The listing of the Designed Properties shows the corresponding changes in the properties and this is particularly useful for the beam properties where the thickness change in the HAT cross-section produces changes in the properties that are not a linear function of the design variable.

----- DESIGNED PROPERTIES -----

PROPERTY TYPE	PROPERTY ID	PROPERTY NAME	TYPE OF PROPERTY	LOWER BOUND	INPUT VALUE	OUTPUT VALUE	UPPER BOUND
PSHELL	1	T	DVPROLL1	1.0000E-02	1.5000E-01	1.0851E-01	1.0000E+20
PBARL	3	DIM2	DVPROLL1	N/A	1.0000E-01	8.4327E-02	N/A
PBAR	3	A	SEC PRO	1.0000E-15	9.6000E-01	8.1218E-01	1.0000E+20
PBAR	3	I1	SEC PRO	1.0000E-15	1.2072E+00	1.0304E+00	1.0000E+20
PBAR	3	I2	SEC PRO	1.0000E-15	9.8120E-01	8.3704E-01	1.0000E+20
PBAR	3	J	SEC PRO	1.0000E-15	3.2234E-03	1.9374E-03	1.0000E+20
PBAR	3	C1	SEC PRO	N/A	1.5000E+00	1.4953E+00	N/A
PBAR	3	D1	SEC PRO	N/A	-1.5000E+00	-1.5047E+00	N/A
PBAR	3	E1	SEC PRO	N/A	-1.5000E+00	-1.5047E+00	N/A
PBAR	3	F1	SEC PRO	N/A	1.5000E+00	1.4953E+00	N/A
PBAR	3	K1	SEC PRO	N/A	5.4779E-01	5.4799E-01	N/A
PBAR	3	K2	SEC PRO	N/A	1.2003E-01	1.1983E-01	N/A

----- DESIGNED CONNECTIVITY PROPERTIES -----

ELEMENT TYPE	ELEMENT ID	PROPERTY NAME	TYPE OF PROPERTY	LOWER BOUND	INPUT VALUE	OUTPUT VALUE	UPPER BOUND
CBAR	31	W3A	DVCRELL1	N/A	1.5750E+00	1.5543E+00	N/A
CBAR	32	W3A	DVCRELL1	N/A	1.5750E+00	1.5543E+00	N/A
CBAR	33	W3A	DVCRELL1	N/A	1.5750E+00	1.5543E+00	N/A
CBAR	34	W3A	DVCRELL1	N/A	1.5750E+00	1.5543E+00	N/A
CBAR	31	W3B	DVCRELL1	N/A	1.5750E+00	1.5543E+00	N/A
CBAR	32	W3B	DVCRELL1	N/A	1.5750E+00	1.5543E+00	N/A
CBAR	33	W3B	DVCRELL1	N/A	1.5750E+00	1.5543E+00	N/A
CBAR	34	W3B	DVCRELL1	N/A	1.5750E+00	1.5543E+00	N/A

The listing of Constraints on Responses shows that the maximum constraint value numerically zero (on the output as -6.49e-5).

----- DESIGN CONSTRAINTS ON RESPONSES -----

(MAXIMUM RESPONSE CONSTRAINTS MARKED WITH **)

INTERNAL ID	DCONSTR ID	INTERNAL RESPONSE ID	EXTERNAL DRESPX ID	RESPONSE TYPE	L/U FLAG	INTERNAL REGION ID	SUBCASE ID	INPUT VALUE	OUTPUT VALUE
1	10	2	3	STRESS	UPPER	2	1	-4.3148E-01	-4.2829E-02
2	10	3	3	STRESS	UPPER	2	1	-3.9778E-01	-6.4857E-05
3	10	4	3	STRESS	UPPER	2	1	-4.3148E-01	-4.2829E-02
4	10	5	3	STRESS	UPPER	2	1	-3.9778E-01	-6.4857E-05**
5	10	6	6	STRESS	UPPER	2	1	-4.6689E-01	-9.0600E-02
6	10	7	6	STRESS	UPPER	2	1	-4.5410E-01	-8.1339E-02
7	10	8	6	STRESS	UPPER	2	1	-4.6689E-01	-9.0600E-02
8	10	9	6	STRESS	UPPER	2	1	-4.5410E-01	-8.1339E-02
9	10	11	1	STRESS	UPPER	1	2	-1.6392E-01**	-7.3114E-04
10	10	12	2	STRESS	UPPER	1	2	-4.9124E-01	-3.8934E-01
11	10	13	2	STRESS	UPPER	1	2	-2.2041E-01	-6.4889E-02
12	30	10	14	DISP	UPPER	14	2	-4.9579E-01	-3.8511E-01



The listing of the Design Responses also shows the input and output values; note that two responses are now at their limits. It is important to note that the constraints and responses in the preceding output are only approximate, because they are based on the approximate model (see [Type-1 Response Evaluation](#)).

RESPONSES IN DESIGN MODEL									
(N/A = BOUND NOT ACTIVE OR AVAILABLE) (** VIOLENT RESPONSES MARKED WITH V ***) (** ACTIVE RESPONSES MARKED WITH A ***)									
----- WEIGHT RESPONSE -----									
INTERNAL ID	DRESP1 ID	RESPONSE LABEL	ROW ID	COLUMN ID	LOWER BOUND	INPUT VALUE	OUTPUT VALUE		UPPER BOUND
1	15	W	3	3	N/A	6.9618E+00	5.3694E+00		N/A
DESIGN CYCLE = 1 SUBCASE = 1									
----- STRESS RESPONSES -----									
INTERNAL ID	DRESP1 ID	RESPONSE LABEL	ELEMENT ID	VIEW ELM ID	COMPONENT NO.	LOWER BOUND	INPUT VALUE	OUTPUT VALUE	UPPER BOUND
2	3	S13	2		9	N/A	1.4213E+04	2.3929E+04	2.5000E+04
3	3	S13		3	9	N/A	1.5056E+04	2.4998E+04	2.5000E+04 A
4	3	S13	14		9	N/A	1.4213E+04	2.3929E+04	2.5000E+04
5	3	S13	15		9	N/A	1.5056E+04	2.4998E+04	2.5000E+04 A
6	6	S16	2		17	N/A	1.3328E+04	2.2735E+04	2.5000E+04
7	6	S16	3		17	N/A	1.3648E+04	2.2967E+04	2.5000E+04
8	6	S16	14		17	N/A	1.3328E+04	2.2735E+04	2.5000E+04
9	6	S16	15		17	N/A	1.3648E+04	2.2967E+04	2.5000E+04
DESIGN CYCLE = 1 SUBCASE = 2									
----- DISPLACEMENT RESPONSES -----									
INTERNAL ID	DRESP1 ID	RESPONSE LABEL	GRID ID	COMPONENT NO.	LOWER BOUND	INPUT VALUE	OUTPUT VALUE		UPPER BOUND
10	14	D2	10203	3	N/A	1.5126E-02	1.8447E-02		3.0000E-02
----- STRESS RESPONSES -----									
INTERNAL ID	DRESP1 ID	RESPONSE LABEL	ELEMENT ID	VIEW ELM ID	COMPONENT NO.	LOWER BOUND	INPUT VALUE	OUTPUT VALUE	UPPER BOUND
12	2	SBARB	32		14	N/A	1.2719E+04	1.5266E+04	2.5000E+04
13	2	SBARB	33		14	N/A	1.9490E+04	2.3378E+04	2.5000E+04

The optimizer has been able to reduce the objective without violating any of the response limits. 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. The output Design Variable and Designed Property values are exact and are not changed by the finite element analysis.

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)
*****
MAXIMUM OF RELATIVE PROP. CHANGES      3.9898E-01 MUST BE LESS THAN
1.0000E-03
--- AND ---
MAXIMUM OF RELATIVE D.V. CHANGES      2.7659E-01 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:

```
^ ^ ^ USER INFORMATION MESSAGE 9051 (FEA)
^ ^ ^ STATIC ANALYSIS INITIATED. 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. Even though convergence has not yet been achieved, the test results are reported nonetheless.

```
CONVERGENCE NOT ACHIEVED YET (HARD CONVERGENCE DECISION LOGIC)
-----
OR      RELATIVE CHANGE IN OBJECTIVE      : 2.2874E-01 MUST BE LESS THAN 1.0000E-03
        ABSOLUTE CHANGE IN OBJECTIVE     : 1.5924E+00 MUST BE LESS THAN 1.0000E-20
        --- AND ---
        MAXIMUM CONSTRAINT VALUE       : 7.1964E-02 MUST BE LESS THAN 5.0000E-03
        (CONVERGENCE TO A FEASIBLE DESIGN)
        --- OR ---
AND      MAXIMUM OF RELATIVE PROP. CHANGES : 3.9898E-01 MUST BE LESS THAN 1.0000E-03
        MAXIMUM OF RELATIVE D.V. CHANGES : 2.7659E-01 MUST BE LESS THAN 1.0000E-03
        (CONVERGENCE TO A BEST COMPROMISE INFEASIBLE DESIGN)
```

Note that convergence has failed on all counts. The objective has changed more than is permitted, the maximum constraint (based on the exact analysis) is violated and the design variables and properties have changed significantly. This is not surprising for the first redesign task.

Insight into the validity of the approximate model can be obtained by presenting some of the design optimization results from the second design cycle.

```
*****
*
*
*           DESIGN OPTIMIZATION
*
*****
*
```

```
*****
*
*           DESIGN CYCLE    2
*
*****
```



----- DESIGN CONSTRAINTS ON RESPONSES -----

(MAXIMUM RESPONSE CONSTRAINTS MARKED WITH **)

INTERNAL ID	DCONSTR ID	INTERNAL RESPONSE ID	EXTERNAL DRESPX ID	RESPONSE TYPE	L/U FLAG	INTERNAL REGION ID	SUBCASE ID	INPUT VALUE	OUTPUT VALUE
1	10	3	3	STRESS	UPPER	2	1	2.9827E-02	-4.5212E-02
2	10	4	3	STRESS	UPPER	2	1	7.1964E-02	-4.4486E-03**
3	10	5	3	STRESS	UPPER	2	1	-3.9299E-01	-4.2598E-01
4	10	6	3	STRESS	UPPER	2	1	2.9827E-02	-4.5212E-02
5	10	7	3	STRESS	UPPER	2	1	7.1964E-02**	-4.4486E-03
6	10	8	3	STRESS	UPPER	2	1	-3.9299E-01	-4.2598E-01
7	10	9	6	STRESS	UPPER	2	1	-4.2040E-01	-4.5636E-01
8	10	10	6	STRESS	UPPER	2	1	-1.7585E-02	9.1286E-02
9	10	11	6	STRESS	UPPER	2	1	-9.2282E-03	-8.2167E-02
10	10	12	6	STRESS	UPPER	2	1	-3.1864E-01	-3.5641E-01
11	10	13	6	STRESS	UPPER	2	1	-4.4727E-01	-4.8071E-01
12	10	14	6	STRESS	UPPER	2	1	-4.0146E-01	-4.4015E-01
13	10	15	6	STRESS	UPPER	2	1	-4.4727E-01	-4.8071E-01
14	10	16	6	STRESS	UPPER	2	1	-4.0146E-01	-4.4015E-01
15	10	17	6	STRESS	UPPER	2	1	-4.2040E-01	-4.5636E-01
16	10	18	6	STRESS	UPPER	2	1	-1.7585E-02	9.1286E-02
17	10	19	6	STRESS	UPPER	2	1	-9.2282E-03	-8.2167E-02
18	10	20	6	STRESS	UPPER	2	1	-3.1864E-01	-3.5641E-01
19	20	2	13	DISP	UPPER	13	1	3.3605E-02	-8.5977E-02
20	10	22	1	STRESS	UPPER	1	2	-4.2589E-01	-4.2811E-01
21	10	23	1	STRESS	UPPER	1	2	-4.1792E-03	-5.6009E-03
22	10	24	2	STRESS	UPPER	1	2	-3.9115E-01	-3.9247E-01
23	10	25	2	STRESS	UPPER	1	2	-6.7333E-02	-6.9596E-02
24	30	21	14	DISP	UPPER	14	2	-3.8900E-01	-3.9111E-01

| R E S P O N S E S I N D E S I G N M O D E L |

(N/A - BOUND NOT ACTIVE OR AVAILABLE)
(***) VIOLATED RESPONSES MARKED WITH V (***)
(***) ACTIVE RESPONSES MARKED WITH A (***)

----- WEIGHT RESPONSE -----

INTERNAL ID	DRESP1 ID	RESPONSE LABEL	ROW ID	COLUMN ID	LOWER BOUND	INPUT VALUE	OUTPUT VALUE	UPPER BOUND
1	15	W	3	3	N/A	5.3694E+00	5.4910E+00	N/A

DESIGN CYCLE = 2 SUBCASE = 1

----- DISPLACEMENT RESPONSES -----

INTERNAL ID	DRESP1 ID	RESPONSE LABEL	GRID ID	COMPONENT NO.	LOWER BOUND	INPUT VALUE	OUTPUT VALUE	UPPER BOUND
2	13	D1	10302	3	N/A	1.0336E-01	9.1402E-02	1.0000E-01

----- STRESS RESPONSES -----

INTERNAL ID	DRESP1 ID	RESPONSE LABEL	ELEMENT ID	VIEW ELM ID	COMPONENT NO.	LOWER BOUND	INPUT VALUE	OUTPUT VALUE	UPPER BOUND
3	3	S13	2	9	N/A	2.5746E+04	2.3870E+04	2.5000E+04	
4	3	S13	3	9	N/A	2.6799E+04	2.4889E+04	2.5000E+04	A
5	3	S13	4	9	N/A	1.5175E+04	1.4350E+04	2.5000E+04	
6	3	S13	14	9	N/A	2.5746E+04	2.3870E+04	2.5000E+04	
7	3	S13	15	9	N/A	2.6799E+04	2.4889E+04	2.5000E+04	
8	3	S13	16	9	N/A	1.5175E+04	1.4350E+04	2.5000E+04	
9	6	S16	1	17	N/A	1.4490E+04	1.3591E+04	2.5000E+04	
10	6	S16	2	17	N/A	2.4560E+04	2.2718E+04	2.5000E+04	
11	6	S16	3	17	N/A	2.4769E+04	2.2946E+04	2.5000E+04	
12	6	S16	4	17	N/A	1.7034E+04	1.6090E+04	2.5000E+04	
13	6	S16	5	17	N/A	1.3818E+04	1.2982E+04	2.5000E+04	
14	6	S16	8	17	N/A	1.4964E+04	1.3996E+04	2.5000E+04	
15	6	S16	9	17	N/A	1.3818E+04	1.2982E+04	2.5000E+04	
16	6	S16	12	17	N/A	1.4964E+04	1.3996E+04	2.5000E+04	
17	6	S16	13	17	N/A	1.4490E+04	1.3591E+04	2.5000E+04	
18	6	S16	14	17	N/A	2.4560E+04	2.2718E+04	2.5000E+04	
19	6	S16	15	17	N/A	2.4769E+04	2.2946E+04	2.5000E+04	
20	6	S16	16	17	N/A	1.7034E+04	1.6090E+04	2.5000E+04	

DESIGN CYCLE = 2 SUBCASE = 2

----- DISPLACEMENT RESPONSES -----



INTERNAL ID	DRESP1 ID	RESPONSE LABEL	GRID ID	COMPONENT NO.	LOWER BOUND	INPUT VALUE	OUTPUT VALUE	UPPER BOUND
21	14	D2	10203	3	N/A	1.8330E-02	1.8267E-02	3.0000E-02
----- STRESS RESPONSES -----								
INTERNAL ID	DRESP1 ID	RESPONSE LABEL	ELEMENT ID	VIEW ELM ID	COMPONENT NO.	LOWER BOUND	INPUT VALUE	UPPER BOUND

22	1	SBARA	33		7	N/A	1.4353E+04	1.4297E+04
23		SBARA	34		7	N/A	2.4896E+04	2.4860E+04
24	2	SBARB	32		14	N/A	1.5221E+04	1.5188E+04
25		SBARB	33		14	N/A	2.3317E+04	2.3260E+04

A review of the DESIGN CONSTRAINTS ON RESPONSES shows a constraint with a positive value when the approximate model results for the first design cycle indicated that all constraints have been satisfied. The maximum constraint violation from the exact analysis is only 7.2% so the approximate values were not that far off. Further comparisons of the RESPONSES IN THE DESIGN MODEL from the two design cycles gives further information on the validity of the approximate model. The output WEIGHT response from the first design cycle (5.3694) is identical with the input for the second design cycle. This should not be too surprising since the weight is a linear function of the design variables in this case so there should not be any approximation in this response. The grid and element responses differ and it is seen, in fact, that the set of retained responses differ somewhat for the two design cycles. A response that is retained in both design cycles is the von Mises stress at the top center (item code 9) of CQUAD4 element with ID=15. For the first design cycle, this is the fifth internal response and the output value is 25,000. At the second design cycle, this is the seventh internal response and the input value is 26,799. This is a fairly significant discrepancy, but recall that the thickness of the plate at which the stress is being measured has been changed from 0.15000 to 0.10852.

This is an appropriate place to show the use of DOPTPRM parameters that can affect the prints that occur based on the P1 and P2 parameters. For this simple case, the amount of print is modest, but one can imagine larger models could produce an overwhelming amount of printout. Suppose the DOPTPRM entry at the end of Listing 8-8 is replaced with:

```
$
$...Optional override of optimization parameters:
$DOPTPRM IPRINT    7      DESMAX   20      DELP     0.5      P1       1      +
+      P2      15      p2cr     5      p2rset   10
set1    10      1      2
```

The P2CR parameter is now requesting that only the five most critical constraints on response quantities are to be printed while the P2RSET = 10 points to a SET1 entry that indicates that only retained responses for DRESP1 IDs 1 and 2 are to be printed. The results from the two parameters are shown:

----- DESIGN CONSTRAINTS ON RESPONSES -----										
(MAXIMUM RESPONSE CONSTRAINTS MARKED WITH **)										
INTERNAL ID	DCONSTR ID	INTERNAL RESPONSE ID	EXTERNAL DRESPX ID	RESPONSE TYPE	L/U FLAG	INTERNAL REGION ID	SUBCASE ID	INPUT VALUE	OUTPUT VALUE	
5	10	7	3	STRESS	UPPER	2	1	7.1964E-02**	-4.4486E-03	
2	10	4	3	STRESS	UPPER	2	1	7.1964E-02	-4.4486E-03**	
19	20	2	13	DISP	UPPER	13	1	3.3605E-02	-8.5977E-02	



4 1	10 10	6 3	3 3	STRESS STRESS	UPPER UPPER	2 2	1 1	2.9827E-02 2.9827E-02	-4.5212E-02 -4.5212E-02
--------	----------	--------	--------	------------------	----------------	--------	--------	--------------------------	----------------------------

| R E S P O N S E S I N D E S I G N M O D E L |

(N/A - BOUND NOT ACTIVE OR AVAILABLE)
 (*** VIOLATED RESPONSES MARKED WITH V ***)
 (*** ACTIVE RESPONSES MARKED WITH A ***)

D E S I G N C Y C L E = 2 S U B C A S E = 2
 ----- STRESS RESPONSES -----

INTERNAL ID	DRESP1 ID	RESPONSE LABEL	ELEMENT ID	VIEW ELM ID	COMPONENT NO.	LOWER BOUND	INPUT VALUE	OUTPUT VALUE	UPPER BOUND
22	1	SBARA	33		7	N/A	1.4353E+04	1.4297E+04	2.5000E+04
23	1	SBARA	34		7	N/A	2.4896E+04	2.4860E+04	2.5000E+04
24	2	SBARB	32		14	N/A	1.5221E+04	1.5188E+04	2.5000E+04
25	2	SBARB	33		14	N/A	2.3317E+04	2.3260E+04	2.5000E+04

As requested, the five largest response constraints are printed and they have been sorted based on the input (to the approximate optimization task) constraint values with the most critical occurring first. For the design responses, the four retained responses associated with DRESP1 = 1 or 2 are printed.

Continuing test problem DSOUKG4, the design cycle proceeds for three design cycles when the following soft convergence results are presented:

```
*****
TERMINATION OF DESIGN ITERATION (SOFT CONVERGENCE)
*****
MAXIMUM OF RELATIVE PROP. CHANGES 1.2290E-16 MUST BE LESS THAN 1.0000E-03
--- AND ---
MAXIMUM OF RELATIVE D.V. CHANGES 1.2290E-16 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.
```

In this case, there is almost no change in the design and the optimization task could be terminated at this point. However, since the SOFTEXIT parameter is set to the default value of "NO" the run continues to hard convergence.

```
*****
NORMAL CONVERGENCE CRITERIA SATISFIED ***** (HARD CONVERGENCE DECISION LOGIC)
*****
CONVERGENCE ACHIEVED BASED ON THE FOLLOWING CRITERIA
(HARD CONVERGENCE DECISION LOGIC)

OR RELATIVE CHANGE IN OBJECTIVE 0.0000E+00 MUST BE LESS THAN 1.0000E-03
ABSOLUTE CHANGE IN OBJECTIVE 0.0000E+00 MUST BE LESS THAN 1.0000E-20
--- AND ---
MAXIMUM CONSTRAINT VALUE -2.9793E-03 MUST BE LESS THAN 5.0000E-03
(Convergence to a feasible design)
--- OR ---
MAXIMUM OF RELATIVE PROP. CHANGES 1.2290E-16 MUST BE LESS THAN 1.0000E-03
AND MAXIMUM OF RELATIVE D.V. CHANGES 1.2290E-16 MUST BE LESS THAN 1.0000E-03
(Convergence to a best compromise infeasible design)
*****
```

In this case, the relative change in the objective is zero and the maximum constraint value is less than the criterion so convergence has been achieved.

Following convergence, additional data recovery may occur and final design optimization data, as indicated in the following abstracted fragments, are printed as governed by the P2 parameter. In this case,



it is necessary to only print the values from the final analysis since that is the only new information that is available (approximate model results have already been printed).

```
*****
*
*
*          D E S I G N      O P T I M I Z A T I O N
*
*
*****
*****
```



```
*****
*
*          F I N A L      A N A L Y S I S
*
*****
*****
```



```
*****      ANALYSIS RESULTS BASED ON THE FINAL DESIGN      *****
----- DESIGN OBJECTIVE -----

```

INTERNAL RESPONSE ID	DRESPX	RESPONSE TYPE	MINIMIZE OR MAXIMIZE	SUPERELEMENT ID	SUBCASE ID	VALUE
1	DRESP1	WEIGHT	MINIMIZE	0	0	5.4910E+00


```
----- DESIGN VARIABLES -----

```

INTERNAL ID	DESVAR ID	LABEL	LOWER BOUND	VALUE	UPPER BOUND
1	1	T-PLATE	1.0000E-03	1.1292E-01	1.0000E+01
2	2	HATDIM2	1.0000E-03	8.4210E-01	1.0000E+01


```
----- DESIGN CONSTRAINTS ON RESPONSES -----
(MAXIMUM RESPONSE CONSTRAINTS MARKED WITH **)
```


INTERNAL ID	DCONSTR ID	INTERNAL RESPONSE ID	EXTERNAL DRESPX ID	RESPONSE TYPE	L/U FLAG	INTERNAL REGION ID	SUBCASE ID	VALUE
1	10	3	3	STRESS	UPPER	2	1	-4.3742E-02
2	10	4	3	STRESS	UPPER	2	1	-2.9793E-03
3	10	5	3	STRESS	UPPER	2	1	-4.2546E-01
4	10	6	3	STRESS	UPPER	2	1	-4.3742E-02
5	10	7	3	STRESS	UPPER	2	1	-2.9793E-03**
6	10	8	3	STRESS	UPPER	2	1	-4.2546E-01
7	10	9	6	STRESS	UPPER	2	1	-4.5575E-01
8	10	10	6	STRESS	UPPER	2	1	-8.9797E-02
9	10	11	6	STRESS	UPPER	2	1	-8.0696E-02
10	10	12	6	STRESS	UPPER	2	1	-3.5588E-01
11	10	13	6	STRESS	UPPER	2	1	-4.8016E-01
12	10	14	6	STRESS	UPPER	2	1	-4.3951E-01
13	10	15	6	STRESS	UPPER	2	1	-4.8016E-01
14	10	16	6	STRESS	UPPER	2	1	-4.3951E-01
15	10	17	6	STRESS	UPPER	2	1	-4.5575E-01
16	10	18	6	STRESS	UPPER	2	1	-8.9797E-02
17	10	19	6	STRESS	UPPER	2	1	-8.0696E-02
18	10	20	6	STRESS	UPPER	2	1	-3.5588E-01
19	20	2	13	DISP	UPPER	13	1	-8.1341E-02
20	10	22	1	STRESS	UPPER	1	2	-4.2816E-01
21	10	23	1	STRESS	UPPER	1	2	-5.6958E-03
22	10	24	2	STRESS	UPPER	1	2	-3.9252E-01
23	10	25	2	STRESS	UPPER	1	2	-6.9684E-02
24	30	21	14	DISP	UPPER	14	2	-3.9120E-01


```
----- RESPONSES IN DESIGN MODEL -----
|             R E S P O N S E S   I N   D E S I G N   M O D E L   |

```


(N/A - BOUND NOT ACTIVE OR AVAILABLE)
 (***) VIOLATED RESPONSES MARKED WITH V (***)
 (*** ACTIVE RESPONSES MARKED WITH A ***)


```
----- WEIGHT RESPONSE -----

```

INTERNAL ID	DRESP1	RESPONSE	ROW	COLUMN	LOWER	UPPER
----------------	--------	----------	-----	--------	-------	-------



ID	ID	LABEL	ID	ID	BOUND	VALUE	BOUND
1	15	W	3	3	N/A	5.4910E+00	N/A
FINAL ANALYSIS SUBCASE = 1							
----- DISPLACEMENT RESPONSES -----							
INTERNAL ID	DRESP1 ID	RESPONSE LABEL	GRID ID	COMPONENT NO.	LOWER BOUND	VALUE	UPPER BOUND
2	13	D1	10302	3	N/A	9.1866E-02	1.0000E-01
----- STRESS RESPONSES -----							
INTERNAL ID	DRESP1 ID	RESPONSE LABEL	ELEMENT ID	VIEW ELM ID	COMPONENT NO.	LOWER BOUND	UPPER BOUND
3	3	S13	2		9	N/A	2.3906E+04
4	3	S13	3		9	N/A	2.4926E+04
5	3	S13	4		9	N/A	1.4363E+04
6	3	S13	14		9	N/A	2.3906E+04
7	3	S13	15		9	N/A	2.4926E+04
8	3	S13	16		9	N/A	1.4363E+04
9	6	S16	1		17	N/A	1.3606E+04
10	6	S16	2		17	N/A	2.2755E+04
11	6	S16	3		17	N/A	2.2983E+04
12	6	S16	4		17	N/A	1.6103E+04
13	6	S16	5		17	N/A	1.2996E+04
14	6	S16	8		17	N/A	1.4012E+04
15	6	S16	9		17	N/A	1.2996E+04
16	6	S16	12		17	N/A	1.4012E+04
17	6	S16	13		17	N/A	1.3606E+04
18	6	S16	14		17	N/A	2.2755E+04
19	6	S16	15		17	N/A	2.2983E+04
20	6	S16	16		17	N/A	1.6103E+04
----- DISPLACEMENT RESPONSES -----							
INTERNAL ID	DRESP1 ID	RESPONSE LABEL	GRID ID	COMPONENT NO.	LOWER BOUND	VALUE	UPPER BOUND
21	14	D2	10203	3	N/A	1.8264E-02	3.0000E-02
----- STRESS RESPONSES -----							
INTERNAL ID	DRESP1 ID	RESPONSE LABEL	ELEMENT ID	VIEW ELM ID	COMPONENT NO.	LOWER BOUND	UPPER BOUND
22	1	SBARA	33		7	N/A	1.4296E+04
23	1	SBARA	34		7	N/A	2.4858E+04
24	2	SBARB	32		14	N/A	1.5187E+04
25	2	SBARB	33		14	N/A	2.3258E+04

After this final design cycle print, the summary of the design cycle history is given:

```
*****
SUMMARY OF DESIGN CYCLE HISTORY
*****
(HARD CONVERGENCE ACHIEVED)
(SOFT CONVERGENCE ACHIEVED)
```

NUMBER OF FINITE ELEMENT ANALYSES COMPLETED 4
 NUMBER OF OPTIMIZATIONS W.R.T. APPROXIMATE MODELS 3

OBJECTIVE AND MAXIMUM CONSTRAINT HISTORY				
CYCLE NUMBER	OBJECTIVE FROM APPROXIMATE OPTIMIZATION	OBJECTIVE FROM EXACT ANALYSIS	FRACTIONAL ERROR OF APPROXIMATION	MAXIMUM VALUE OF CONSTRAINT
INITIAL		6.961800E+00		-1.639152E-01
1	5.369376E+00	5.369376E+00	3.187558E-13	7.196367E-02
2	5.490995E+00	5.490995E+00	2.183650E-14	-2.979310E-03
3	5.490995E+00	5.490995E+00	0.000000E+00	-2.979310E-03



DESIGN VARIABLE HISTORY											
INTERNAL DV. ID.	EXTERNAL DV. ID.	LABEL	INITIAL	:	1	:	2	:	3	:	4
5	:										
1	1	T-PLATE	1.5000E-01	:	1.0851E-01	:	1.1292E-01	:	1.1292E-01	:	
2	2	HATDIM2	1.0000E+00	:	8.4327E-01	:	8.4210E-01	:	8.4210E-01	:	

From this summary we see that an optimal design was found after 3 design cycles and 4 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 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 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.

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 is small, errors in constraints can be much greater.

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.)

A final message is printed which indicates why the optimization task was stopped. In this case, this message says:

```
*** USER INFORMATION MESSAGE 6464 (DOM12E)
RUN TERMINATED DUE TO HARD CONVERGENCE TO AN OPTIMUM AT CYCLE NUMBER =
```

3.

indicating that the design task was successfully completed.

Other possible messages for the end of the optimization task and a brief explanation for each are:

- “RUN TERMINATED DUE TO MAXIMUM NUMBER OF DESIGN CYCLES =”
The user prescribed maximum number of design cycles FSDMAX + DESMAX have been carried out without achieving convergence.
- “RUN TERMINATED DUE TO PARAMETER OPTEXIT = n”
The OPTEXIT parameter permits exiting the design tasks at a number of points.
- “RUN TERMINATED DUE TO SOFT CONVERGENCE AT CYCLE NUMBER =” Parameter SOFTEXIT has been set to YES and soft convergence was achieved



- “RUN TERMINATED DUE TO HARD CONVERGENCE TO A BEST COMPROMISE INFEASIBLE DESIGN AT CYCLE NUMBER = “
The optimizer could not find a feasible design and it could make no further progress, therefore the run was terminated (see [Compromise Infeasible Design](#).)
- “RUN TERMINATED DUE TO CASE CONTROL COMMAND DSAPRT(END=SENSITIVITY) “
The DSAPRT command has requested a design sensitivity only run.
- “RUN TERMINATED DUE TO POOR GEOMETRY OR POOR PROPERTIES. SEE FATAL MESSAGES PRINTED ABOVE FOR INFORMATION”
The redesign has produced a finite element model that violates finite element modeling rules. For example, a mesh has distorted elements or an element property has taken on a value that is not allowed, such as a negative shell thickness.
- “RUN TERMINATED DUE TO MODE TRACKING FAILURE. SEE USER FATAL MESSAGE 6677 (MTFRD) PRINTED ABOVE “
The mode tracking algorithm (see [Mode Tracking](#)) was unable to do its job and the run was therefore terminated.

Modification of Move Limit Parameters

As explained in the [Automatic Updates of Move Limits](#), the optimization algorithm sometimes reduces the move limit parameters by a factor of 2.0. When this occurs, a message like the following appears in the .f06 file:

```
*****
*                               USER      WARNING      MESSAGE
*
* IF YOU WANT TO CONTINUE THE DESIGN AFTER THIS JOB
* IS COMPLETED, YOU MUST INCLUDE A REVISED DOPTPRM
* BULK DATA ENTRY IN THE BULK DATA SECTION WITH THE
* FOLLOWING ITEMS MODIFIED AS SHOWN:
*
*          DELP = 1.0000E-01
*          DPMIN = 5.0000E-03
*          DELX = 5.0000E-01
*          DXMIN = 2.5000E-02
*
*****
*
* IF A DELXV IS SPECIFIED ON A DESVAR BULK DATA ENTRY,
* UPDATED DELXV VALUES ARE PRESENT ON THE DESVAR
* ENTRIES CONTAINED IN THE PUNCH FILE.
*
*
* (NOTE: THERE MAY BE MORE THAN ONE MESSAGE LIKE
* THIS. THE LAST ONE IN THIS RUN SHOULD
* BE LOCATED AND USED.)
*
*****
```

This message never appeared in the DSOUG4.F06 output discussed in this section, but still requires explanation. It is seen that four move limits have been revised and guidance is provided for continuing the design task in a new run using the final design variable values obtained in the current run. There are several guidelines contained in the above listing. The first is that the new values should be used if the



design task is continued. The thinking is that the original move limits were too loose, so the reduced values are preferred. The statement that “YOU MUST INCLUDE A REVISED...” is somewhat misleading in that the new job will run with the original values for the parameters. Changing these limits is really left to the user’s judgment. The second guideline points out that DELXV values included on DESVAR entries in the punch file (see [Design Punch Output](#)) reflect the reduction in the individual move limits. It may be that these move limits are now too restrictive and it will be necessary to manually adjust the DELXV values on the DESVAR entries.

Special Prints for Fully Stressed Design

The output produced by the FSD algorithm is very similar to that just presented for the Mathematical Programming (MP) algorithm. One significant difference is in the results that are printed at each design cycle. With MP, these results are printed after an approximate optimization and before a reanalysis has occurred. With FSD, there is no approximate analysis because the optimizer is never called. The output is deferred until a new analysis has been performed. This is indicated in the output by the printing of the following message:

```
*****OPTIMIZATION RESULTS BASED ON AN EXACT ANALYSIS *****
```

instead of the message:

```
*****OPTIMIZATION RESULTS BASED ON THE APPROXIMATE MODEL****
```

The absence of approximate results also affects the SUMMARY OF THE DESIGN CYCLE HISTORY that occurs at the end of the run. The following printout shows this table resulting from a design task that included five FSD cycles followed by MP cycles to convergence. Note that the number of FSD cycles appears at the top of the table, and that no approximate objective values and therefore no “fractional error” are output for the FSD cycles.

SUMMARY OF DESIGN CYCLE HISTORY				
(HARD CONVERGENCE ACHIEVED)				
	NUMBER OF FINITE ELEMENT ANALYSES COMPLETED	10		
	NUMBER OF FULLY STRESSED DESIGN CYCLES COMPLETED	5		
	NUMBER OF OPTIMIZATIONS W.R.T. APPROXIMATE MODELS	4		
OBJECTIVE AND MAXIMUM CONSTRAINT HISTORY				
CYCLE NUMBER	OBJECTIVE FROM APPROXIMATE OPTIMIZATION	OBJECTIVE FROM EXACT ANALYSIS	FRACTIONAL ERROR OF APPROXIMATION	MAXIMUM VALUE OF CONSTRAINT
INITIAL		4.828427E+00		-3.234952E-01
1	FSD	2.668171E+00	N/A	4.203515E-02
2	FSD	2.502887E+00	N/A	8.190254E-02
3	FSD	2.541077E+00	N/A	6.268603E-02
4	FSD	2.593662E+00	N/A	4.396826E-02
5	FSD	2.635597E+00	N/A	3.135078E-02
6	2.709053E+00	2.709045E+00	2.640250E-06	3.502930E-04
7	2.702080E+00	2.702064E+00	6.000030E-06	7.404297E-04
8	2.695884E+00	2.695881E+00	1.061257E-06	2.517090E-03
9	2.694964E+00	2.694966E+00	-7.962130E-07	2.553125E-03



Special Prints for Discrete Variable Optimization

The output produced by the discrete variable processing methods is similar to the continuous optimization results. There is no change for results printed at each continuous optimization design cycle. At each discrete variable processing cycle, the feasibility of a new discrete solution is checked based on both approximate and exact reanalysis, a similar approach to the one taken for continuous variable optimization. The approximate analysis is based on the approximate model and is labeled "Soft Feasible" if all the constraints are satisfied. The results of the exact analysis of the discrete design are labeled "Hard Feasible" if all the constraints are satisfied. "Soft Infeasible" and "Hard Infeasible" indicate that at least one constraint is violated in either the approximate or exact analyses, respectively.

```
***** A SOFT FEASIBLE DISCRETE SOLUTION FOUND (SOFT FEASIBILITY DISCRETE SOLUTION CHECK LOGIC) *****
      MAXIMUM CONSTRAINT VALUE : -3.6252E-02 MUST BE LESS THAN 5.0000E-03
*****
***** A HARD FEASIBLE DISCRETE SOLUTION FOUND (HARD FEASIBILITY DISCRETE SOLUTION CHECK LOGIC) *****
      MAXIMUM CONSTRAINT VALUE : 4.7029E-03 MUST BE LESS THAN 5.0000E-03
*****
```

At each discrete variable optimization cycle, the design objective, design variables, designed properties, and design constraints are presented after the preceding continuous design cycle. For the discrete cycle, there is no input value since the input values are the output values of the preceding continuous design cycle. Discrete results are indicated by the printing of the following message. (Note that these results are based on a full finite element analysis.)

```
*****
*          D I S C R E T E   D E S I G N   C Y C L E       6D   *
******
*****      OPTIMIZATION RESULTS BASED ON AN EXACT ANALYSIS *****
```

The existing "SUMMARY OF DESIGN CYCLE HISTORY" is modified to include these new discrete designs. An example of this new output follows. The summary table also shows the number of discrete processing analyses completed and whether a soft or hard feasible discrete solution is obtained. Note that the cycle numbers flagged with a "D" correspond to the discrete results that have been computed based on the preceding continuous design.

```
***** SUMMARY OF DESIGN CYCLE HISTORY *****
*****
```

(HARD CONVERGENCE ACHIEVED)	
(SOFT CONVERGENCE ACHIEVED)	
(HARD FEASIBLE DISCRETE SOLUTION OBTAINED)	
(SOFT FEASIBLE DISCRETE SOLUTION OBTAINED)	

NUMBER OF FINITE ELEMENT ANALYSES COMPLETED	7
NUMBER OF OPTIMIZATIONS W.R.T. APPROXIMATE MODELS	6
NUMBER OF DISCRETE PROCESSING ANALYSES COMPLETED	2

OBJECTIVE AND MAXIMUM CONSTRAINT HISTORY				
CYCLE NUMBER	OBJECTIVE FROM APPROXIMATE OPTIMIZATION	OBJECTIVE FROM EXACT ANALYSIS	FRACTIONAL ERROR OF APPROXIMATION	MAXIMUM VALUE OF CONSTRAINT
INITIAL		4.740178E+02		2.907700E-02
1	3.957753E+02	3.924499E+02	8.473461E-03	7.261696E-04



2	2.943152E+02	2.943384E+02	-7.900563E-05	7.262860E-04
3	1.962019E+02	1.962255E+02	-1.202970E-04	6.985210E-04
4	9.808932E+01	9.811287E+01	-2.400495E-04	6.985210E-04
5	4.883356E+00	4.905637E+00	-4.541855E-03	6.985210E-04
5D	4.93356E+00	5.056371E+00	-4.38652 E-03	9.285210E-04
6	4.905637E+00	4.905637E+00	0.000000E+00	6.985210E-04
6D	4.92156E+00	5.041347E+00	-4.28652 E-03	7.253256E-04



DESIGN VARIABLE HISTORY

INTERNAL DV. ID.	EXTERNAL DV. ID.	LABEL	INITIAL	: 1	: 2	: 3	: 4	: 5
1	1	LAYER1	1.0000E+00	: 1.2734E+00 : 1.3084E+00 : 1.2966E+00 : 1.2829E+00 : 1.2671E+00				
2	2	E	1.0000E+00	: 9.9815E-01 : 9.9593E-01 : 9.9645E-01 : 1.0015E+00 : 1.0146E+00				
3	3	RHO	1.0000E+00	: 8.0000E-01 : 6.0000E-01 : 4.0000E-01 : 2.0000E-01 : 2.0000E-01				
4	4	NU	1.0000E+00	: 1.2000E+00 : 1.4400E+00 : 1.7280E+00 : 2.0736E+00 : 2.4883E+00				

INTERNAL DV. ID.	EXTERNAL DV. ID.	LABEL	: 5D	6:	6D	:
1	1	LAYER1	: 1.2000E+00 : 1.2671E+00 : 1.2000E+00 :			
2	2	E	: 1.0000E+00 : 1.0146E-01 : 1.0000E-01 :			
3	3	RHO	: 2.0000E+00 : 1.0000E-01 : 1.0000E-01 :			
4	4	NU	: 2.5000E+00 : 1.2886E+00 : 1.5000E+00 :			

A final message is printed that indicates why the discrete optimization task was terminated. For example,

```
*** USER INFORMATION MESSAGE 6464 (DOM12E)
RUN TERMINATED DUE TO HARD CONVERGENCE TO AN OPTIMUM AT CYCLE NUMBER = 15.
AND HARD FEASIBLE DISCRETE DESIGN OBTAINED
```

indicating that the design task was successfully completed due to hard convergence of the corresponding continuous design task and a hard feasible discrete design was found.

Special Prints for Topology Optimization

The majority of the prints for topology optimization follow that for standard optimization. An exception is when design variables are to be printed. Even small topology optimization tasks are likely to have thousands of design variables. Therefore, the “Comparison Between Input Property Variables from Analysis and Design Models” print never occurs and the design variable print that normally occurs if P2 = 1 is suppressed unless P2>8. Similarly, the final Design Variable History print does not occur unless P2 > 8 is specified.



Design Sensitivity Output

[Design Optimization Output](#) has provided an explanation of the output that is produced in an optimization task. It is also possible to obtain output from a sensitivity analysis and that is the topic of this subchapter. Recall from [Design Sensitivity Analysis](#) 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.) As has been mentioned several times, sensitivity information is valuable in its own right since it provides a quantitative indication of how much a design response will change due to a change in a design variable.

The primary means of requesting design sensitivity output is with the DSAPRT Case Control command as detailed in the [Case Control Section](#). This command provides tables of data that allow you to easily identify which response and design variable are associated with the sensitivity. This form of data is referred to as “formatted” sensitivities and can be contrasted with “unformatted” sensitivities that simply provides a matrix print of the DSCM2 matrix that contains the sensitivity coefficients as

$$(DSCM2)_{ij} = \frac{\partial r_j}{\partial x_i} \quad (5-1)$$

where i is the row order and j is the column order.

Unformatted sensitivity data prints can be activated by setting the Bulk Data parameter OPTEXIT to 4 or 7, but this capability has been superseded by the DSAPRT command. All the functionality, and much more, of the OPTEXIT =4 or 7 parameter is available with the DSAPRT command. Both the formatted and unformatted forms of sensitivity data are presented here because the unformatted form has utility in that it provides a more compact form of the data and it is ideal for those who would like to couple MSC Nastran-computed sensitivities with their own postprocessors and/or optimizers.

Formatted Design Sensitivity

Examples of the formatted sensitivity output are presented in the following listings that are based on the DSOU4 example of [Design Optimization Output](#) with the following Case Control command:

DSAPRT(START=1,END=LAST)=ALL

DSAPRT used to provide formatted sensitivity information for the initial and final design.

The first print is for the initial design:

DESIGN SENSITIVITY MATRIX OUTPUT						SEID=0
RESPONSE SENSITIVITY COEFFICIENTS						*
<hr/>						
DRESP1 ID= 15	RESP VALUE	RESPONSE TYPE= WEIGHT	DESIGN VARIABLE	COEFFICIENT	DESIGN VARIABLE	COEFFICIENT
6.9618E+00			1 T-PLATE	2.8300E+01	2 HATDIM2	2.6602E+00
DRESP1 ID= 14		RESPONSE TYPE= DISP	GRID ID=	10203	COMP NO=	3
						SEID=0



SUBCASE	RESP VALUE	DESIGN VARIABLE COEFFICIENT	DESIGN VARIABLE COEFFICIENT
2	1.5126E-02	1 T-PLATE -1.5621E-02	2 HATDIM2 -1.2575E-02
DRESP1 ID= 1	RESPONSE TYPE= STRESS	ELEM ID= 34 COMP NO= 7	SEID=0
SUBCASE RESP VALUE	DESIGN VARIABLE COEFFICIENT	DESIGN VARIABLE COEFFICIENT	
2	2.0902E+04	1 T-PLATE -1.3723E+04	2 HATDIM2 -1.7607E+04
DRESP1 ID= 2	RESPONSE TYPE= STRESS	ELEM ID= 32 COMP NO= 14	SEID=0
SUBCASE RESP VALUE	DESIGN VARIABLE COEFFICIENT	DESIGN VARIABLE COEFFICIENT	
2	1.2719E+04	1 T-PLATE -9.3138E+03	2 HATDIM2 -1.0773E+04
DRESP1 ID= 2	RESPONSE TYPE= STRESS	ELEM ID= 33 COMP NO= 14	SEID=0
SUBCASE RESP VALUE	DESIGN VARIABLE COEFFICIENT	DESIGN VARIABLE COEFFICIENT	
2	1.9490E+04	1 T-PLATE -1.4878E+04	2 HATDIM2 -1.6233E+04
DRESP1 ID= 3	RESPONSE TYPE= STRESS	ELEM ID= 2 COMP NO= 9	SEID=0
SUBCASE RESP VALUE	DESIGN VARIABLE COEFFICIENT	DESIGN VARIABLE COEFFICIENT	
1	1.4213E+04	1 T-PLATE -1.6833E+05	2 HATDIM2 -3.3410E+02
DRESP1 ID= 3	RESPONSE TYPE= STRESS	ELEM ID= 3 COMP NO= 9	SEID=0
SUBCASE RESP VALUE	DESIGN VARIABLE COEFFICIENT	DESIGN VARIABLE COEFFICIENT	
1	1.5056E+04	1 T-PLATE -1.7292E+05	2 HATDIM2 -1.3659E+02
DRESP1 ID= 3	RESPONSE TYPE= STRESS	ELEM ID= 14 COMP NO= 9	SEID=0
SUBCASE RESP VALUE	DESIGN VARIABLE COEFFICIENT	DESIGN VARIABLE COEFFICIENT	
1	1.4213E+04	1 T-PLATE -1.6833E+05	2 HATDIM2 -3.3410E+02
DRESP1 ID= 3	RESPONSE TYPE= STRESS	ELEM ID= 15 COMP NO= 9	SEID=0
SUBCASE RESP VALUE	DESIGN VARIABLE COEFFICIENT	DESIGN VARIABLE COEFFICIENT	
1	1.5056E+04	1 T-PLATE -1.7292E+05	2 HATDIM2 -1.3659E+02
DRESP1 ID= 6	RESPONSE TYPE= STRESS	ELEM ID= 2 COMP NO= 17	SEID=0
SUBCASE RESP VALUE	DESIGN VARIABLE COEFFICIENT	DESIGN VARIABLE COEFFICIENT	
1	1.3328E+04	1 T-PLATE -1.6365E+05	2 HATDIM2 -1.1546E+02
DRESP1 ID= 6	RESPONSE TYPE= STRESS	ELEM ID= 3 COMP NO= 17	SEID=0
SUBCASE RESP VALUE	DESIGN VARIABLE COEFFICIENT	DESIGN VARIABLE COEFFICIENT	
1	1.3648E+04	1 T-PLATE -1.6214E+05	2 HATDIM2 -1.0850E+02
DRESP1 ID= 6	RESPONSE TYPE= STRESS	ELEM ID= 14 COMP NO= 17	SEID=0
SUBCASE RESP VALUE	DESIGN VARIABLE COEFFICIENT	DESIGN VARIABLE COEFFICIENT	
1	1.3328E+04	1 T-PLATE -1.6365E+05	2 HATDIM2 -1.1546E+02
DRESP1 ID= 6	RESPONSE TYPE= STRESS	ELEM ID= 15 COMP NO= 17	SEID=0
SUBCASE RESP VALUE	DESIGN VARIABLE COEFFICIENT	DESIGN VARIABLE COEFFICIENT	
1	1.3648E+04	1 T-PLATE -1.6214E+05	2 HATDIM2 -1.0850E+02

The order of the data for the formatted prints can differ from that given for the RESPONSES IN THE DESIGN MODEL listing since a different sort is used to organize the data. For formatted sensitivities, the order of the data is:



First by superelement ID , then by

Response type, then by

External Response ID, then by

Grid/Element ID, then by

Component, then by

Subcase, then by

Frequency/Time Step

Responses that share the superelement, response type, grid/element ID and component have a single header with the subcase and frequency time/step supplied with the response value. Response ID order corresponds to that given by Table 1 of the DRESP1 Bulk Data entry (see [Bulk Data Entries](#) in the *MSC Nastran Quick Reference Guide*).

The example of the listing does not contain any type2 or type 3 responses. If any are present, they are sorted:

First by superelement ID, then by

External response ID, then by

Subcase, then by

Frequency/time step

The extensive labeling makes the interpretation straightforward, so only a limited amount of explanation is given here. The first response in the table is for RTYPE=WEIGHT. The value of the response is given as 6.9618 and the sensitivity with respect to the first design variable (which has a label T-PLATE) is 28.310. This means that if the value of the first design variable changes by a unit amount, the weight will increase by 28.310. The sensitivity of the weight to the second design variable is 2.6608, indicating that a unit change in this design variable increases the weight by a much smaller amount than the first design variable.



Unformatted Design Sensitivity

Unformatted design sensitivity output consists of two parts: a table listing the column order of the DSCM2 matrix of [Equation \(5-1\)](#), followed by a print of the DSCM2 matrix.

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 independent design variable with the second lowest ID and so on. There are a number of places where this order is specified with the best one perhaps the DESIGN VARIABLE HISTORY summary that was shown at the end of the [DSOUG4 Output](#).

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 *MSC Nastran 2005 r3 DMAP Programmer's Guide* for details.

Design Sensitivity Output Example

This example is taken from the Test Problem Library, problem number D200X5. Modified by the addition of PARAM,OPTEXIT,4 to the Bulk Data Section. Thus, only sensitivity analysis and not optimization will be performed.

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.	Matrix Dimension
OCOLUMN 1	ROWS 1 THRU 3			
ROW 1)	4.0000E+01 2.2500E+01 1.5938E+01			
OCOLUMN 2	ROWS 1 THRU 3			
ROW 2)	3.2871E+01 1.9345E+01 1.3255E+01			
OCOLUMN 3	ROWS 1 THRU 3			
ROW 3)	3.2871E+01 1.9345E+01 1.3255E+01			
OCOLUMN 4	ROWS 1 THRU 3			
ROW 4)	2.5733E+04 2.9702E+04 3.2985E+04			
OCOLUMN 5	ROWS 1 THRU 3			
ROW 5)	2.5733E+04 2.9702E+04 3.2985E+04			
OCOLUMN 6	ROWS 1 THRU 3			
ROW 6)	-1.5881E+05 -1.5696E+05 -1.5544E+05			
OCOLUMN 7	ROWS 1 THRU 3			
ROW 7)	-1.5881E+05 -1.5696E+05 -1.5544E+05			
.	.			
.	.			
.	.			
OCOLUMN 39	ROWS 1 THRU 3			
ROW 39)	3.20435E+05 -2.9491E+04 -5.0688E+03			
OCOLUMN 40	ROWS 1 THRU 3			
ROW 40)	2.1453E+00 1.4776E+00 1.1223E+00			
OCOLUMN 41	ROWS 1 THRU 3			
ROW 41)	2.1453E+00 1.4776E+00 1.1223E+00			
OCOLUMN 42	ROWS 1 THRU 3			
ROW 42)	-1.7913E+04 -1.7736E+04 -1.7589E+04			
OCOLUMN 43	ROWS 1 THRU 3			
ROW 43)	-1.7913E+04 -1.7736E+04 -1.7589E+04			
OCOLUMN 44	ROWS 1 THRU 3			
ROW 44)	-1.7913E+04 -1.7736E+04 -1.7589E+04			
OCOLUMN 45	ROWS 1 THRU 3			
ROW 45)	-1.7913E+04 -1.7736E+04 -1.7589E+04			
.	.			
.	.			
.	.			
OCOLUMN 61	ROWS 1 THRU 3			
ROW 61)	-2.1265E+04 -1.0639E+04 -5.4152E+03			
O THE NUMBER OF NON-ZERO TERMS IN THE DENSEST COLUMN =	3			
O THE DENSITY OF THIS MATRIX IS 100.00 PERCENT.				

$$\text{matrix density} = \left(\frac{\text{rows} \cdot \text{columns} - \text{zero terms}}{\text{rows} \cdot \text{columns}} \right) \cdot 100$$

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 which 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 model 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 NO.	DRESP1 ENTRY ID	RESPONSE TYPE				
1	35	WEIGHT				
----- STATICS RESPONSES -----						
COLUMN NO.	DRESP1 ENTRY ID	RESPONSE TYPE	GRID/ELM ID	COMPONENT NO.	SUB CASE	PLY 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. The responses are sorted by individual response type. This order is given in Table 1 of the DRESP1 Bulk Data entry description (see [Bulk Data Entries](#) in the *MSC Nastran Quick Reference Guide*).



Design Punch Output

The term “punch file” refers to a output file whose origins can now be considered to be associated with the infancy of digital computing. This file contains outputs from the MSC Nastran analysis that can be conveniently used as input to another computer analysis. In the early days of computer analysis, these results could be punched onto computer cards so that they could be read in, via a card reader, as input to another analysis. This may seem quaint now that card punches and card readers are obsolete, but the punch file retains considerable utility for postprocessing of MSC Nastran results. The file is particularly useful in the design sensitivity and optimization context because it produces output in the format of the MSC Nastran bulk data entries. This provides a convenient way of capturing the results of the design process and using these results in a subsequent analysis to either punch fragments of the Bulk Data file or to punch a complete new input Bulk Data file. This section provides an overview of the options that are available for punched output.

Punch Parameters

As briefly explained in [Output-Controlling Parameters](#), the DESPCH parameter indicates how often design Bulk Data information is printed to the punch file while DESPCH1 specifies both the format of the data (small field versus large field) and what data are to be punched. Typically, the default of DESPCH=0 is adequate since this provides information for the final design cycle and it is typically only these data that are of interest in subsequent analyses. Setting DESPCH to a positive integer n produces punched output at every n th design cycle and the last design cycle. This could be useful if a review of the design cycle history indicates that an intermediate design actually is better than the final design. DESPCH can also be set to a negative number, in which case no design data is punched.

The DESPCH1 parameter controls the format and the amount of data that are punched. Positive values of this parameter produce Bulk Data entries in the “large field” format; i.e. each item in the entry occupies 16 spaces. Data in the large field format can be difficult to interpret visually if you are not used to it, but this may be the only way to obtain sufficient numerical accuracy to make the design results usable. Negative values of DESPCH1 produce results in the standard small field (i.e. 8 spaces) format.

The Bulk Data entries that are punched are determined by the absolute value of the DESPCH1 parameter as indicated by [Table 5-1](#).

Table 5-1

DESPCH1	MEANING
0	Nothing is punched
1	Designed property entries (e.g., PROD or MAT1) are punched
2	All properties of a given type are punched when one or more of that property type is designed.
4	Updated DESVAR and DRESP1 entries are punched



Some comments on this table are:

- If any GRID location is being designed, the complete set of GRID points will be punched whenever DESPCH1 is nonzero
- DESPCH1=1 typically produces fewer entries than DESPCH1=2 and is ideal when your bulk data file has the designed properties in a segregated part of the file. DESPCH1=2 supports the case where the designed properties are sprinkled throughout the input file and it is simpler to globally replace all the properties, even those that have not changed
- The DRESP1 punch is for situations where mode tracking has been utilized. The original design model may be imposing a limit on the 3rd mode, but the redesign has placed this physical mode in the 5th position. The punched DRESP1 will indicate that the 5th mode is to be designed. All DRESP1 entries that are of RTYPE=EIGN or RTYPE=FREQ will be punched in this situation.
- Combined quantities can be obtained by summing the three values in the table. For example DESPCH1=6, which is the default, punches all properties, the DESVARs and any DRESP1 entries. Meaningful combined values are 5 and 6. DESPCH1=6 is the default value.

Example for DESPCH

[Figure 5-1](#) is an example of the information found in the .pch based on a DESPCH request when a PCOMP Bulk Data entry has plies that are being designed. The DESIGN CYCLE number is provided in a banner for the file and this is followed by annotations for each type of data that follows. In this case DESVAR entries are provided in the large field format and this is followed by values of the designed PCOMP entry. A feature of PCOMP analysis in MSC Nastran is that the PCOMP data is actually converted to equivalent PSHELL/MAT2 entries. These updated entries are also provided in the .pch file, but are commented out so that the entire file can be included in a bulk data file and there will be no difficulty with duplicate property IDs. In certain applications, it may be desirable to remove the PCOMP data and remove the comments from the PSHELL/MAT2 Bulk Data entries.



```

S ****DESIGN CYCLE NUMBER = 9 ****
S
S UPDATED DESIGN MODEL DATA ENTRIES
S
DESVAR *      1PLY90     8.22658688E-02  1.99999996E-02+D   1V
*D 1V 3.00000000E+02 5.00000000E-01
DESVAR *      2PLYM45    2.85679966E-01  1.99999996E-02+D   2V
*D 2V 3.00000000E+02 5.00000000E-01
DESVAR *      3PLY45     1.99999996E-02  1.99999996E-02+D   3V
*D 3V 3.00000000E+02 5.00000000E-01
DESVAR *      4PLY0      8.38311464E-02  1.99999996E-02+D   4V
*D 4V 3.00000000E+02 5.00000000E-01
S
S UPDATED ANALYSIS MODEL DATA ENTRIES
S
PCOMP*        1 -2.54759588E-03  0.00000000E+00  5.00000000E+03*
*          HILL 0.00000000E+00  0.00000000E+00  SYM*
*          1 4.44235717E-04  9.00000000E+01  YES*
*          1 1.54267182E-03  4.50000000E+01  YES*
*          1 1.08000000E-04  -4.50000000E+01  YES*
*          1 4.52688197E-04  0.00000000E+00  YES
S Spawned PSHELL, MAT2 entries from PCOMP
S
PSHELL*       1 100000001  5.09519130E-03  200000001*
S *          1.00000000E+00  300000001  1.00000000E+00  0.00000000E+00*
S *          -2.54759588E-03  2.54759565E-03  0
S
MAT2*         100000001  8.43634500E+06  3.43898675E+06  2.56559100E+06*
S *          8.37588350E+06  2.56559075E+06  3.73030525E+06  5.09999990E+00*
S *          0.00000000E+00  0.00000000E+00  0.00000000E+00  0.00000000E+00*
S *          0.00000000E+00  0.00000000E+00  0.00000000E+00  0.00000000E+00*
S *          0
S
MAT2*         200000001  4.85598850E+06  3.05657825E+06  2.49240750E+06*
S *          1.27210570E+07  2.49240725E+06  3.34789700E+06  5.09999990E+00*
S *          0.00000000E+00  0.00000000E+00  0.00000000E+00  0.00000000E+00*
S *          0.00000000E+00  0.00000000E+00  0.00000000E+00  0.00000000E+00*
S *          0
S
MAT2*         300000001  7.44932688E+05  0.00000000E+00  0.00000000E+00*
S *          9.02194062E+05  0.00000000E+00  0.00000000E+00  5.09999990E+00*
S *          0.00000000E+00  0.00000000E+00  0.00000000E+00  0.00000000E+00*
S *          0.00000000E+00  0.00000000E+00  0.00000000E+00  0.00000000E+00*
S *          0

```

Figure 5-1 Sample of Punched Output from PARAM DESPCH

Special Punch Considerations for Topology Optimization

The DESPCH parameter has a separate function in Topology Optimization. In this case, PARAM DESPCH n writes output to a *jobname.des* file that lists the element ID and the design variable value for every n-th design cycle. As shown in [Postprocessing Output](#), this file can be brought into Patran for two-dimensional problems to show the topology optimization results. Setting DESPCH 0 (default) produces design variable values for the final design cycle only.

Special Punch in the Case of Shape Optimization

As described above, if shape optimization is being performed, a complete set of GRID Bulk Data entries will be punched every n-th design cycle where n is set using PARAM DESPCH. A special case occurs when the new mesh has become distorted to the extent that the resulting finite elements cannot be generated.

In this case, an invalid mesh message is produced by the EMG module and, if it has not already been punched, the grid data for the last design whose mesh was valid are punched. This allows the user to start up again from this last valid mesh while perhaps changing some move parameters so that the mesh distortions do not occur.



New Bulk Data File

As indicated in the [Case Control Section](#), the ECHO command can be used to cause the writing of a new Bulk Data file after a design task has been completed. [Figure 5-3](#) displays a portion of the file that is produced if the DSOUG4 example of [Design Optimization Output](#) includes the command

```
ECHO = SORT, PUNCH (NEWBULK)
```

A banner indicates the beginning of the updated Bulk Data file. It is apparent that much of the file does not require any change. The first new Bulk Data entries are the CBAR entries that utilize the new beam offset values. Properties and DESVAR entries are also changed, while the responses and constraints do not. It is now very convenient to make additional MSC Nastran runs by including this new Bulk Data file in a new job submittal. The new run may continue the SOL 200 design task or it could perform a completely new type of analysis, such as nonlinear.



```

S ***** *****
S * ENTIRE NEW BULK DATA DECK WITH UPDATED ENTRIES INSERTED *
S * *****
S *****
S BEGIN BULK
PARAM POST -1
PARAM NASPRT 1
$-
$-----ANALYSIS MODEL:-----
$-----
$ GRID 10000 0.0 0.0 0.0
GRID 10001 2.5 0.0 0.0
$-----
$ CQUAD4 1 1 10000 10001 10101 10100
CQUAD4 2 1 10001 10002 10102 10101
$-----
$ CQUAD4 15 1 10302 10303 10403 10402
CQUAD4 16 1 10303 10304 10404 10403
$-
CBAR* 31 3 10200 10201*
* 0.0000000E+00 1.0000000E+00 0.0000000E+00 1*
* 0 0 0.0000000E+00 0.0000000E+00 0.0000000E+00*
* 1.55632222E+00 0.0000000E+00 0.0000000E+00 1.55632222E+00
CBAR* 32 3 10201 10202*
* 0.0000000E+00 1.0000000E+00 0.0000000E+00 1*
* 0 0 0.0000000E+00 0.0000000E+00 0.0000000E+00*
* 1.55632222E+00 0.0000000E+00 0.0000000E+00 1.55632222E+00
CBAR* 33 3 10202 10203*
* 0.0000000E+00 1.0000000E+00 0.0000000E+00 1*
* 0 0 0.0000000E+00 0.0000000E+00 0.0000000E+00*
* 1.55632222E+00 0.0000000E+00 0.0000000E+00 1.55632222E+00
CBAR* 34 3 10203 10204*
* 0.0000000E+00 1.0000000E+00 0.0000000E+00 1*
* 0 0 0.0000000E+00 0.0000000E+00 0.0000000E+00*
* 1.55632222E+00 0.0000000E+00 0.0000000E+00 1.55632222E+00
$-
PSHELL* 1 1 1.12644322E-01 1*
* 1.0000000E+00 0 8.33333313E-01 0.0000000E+00*
* 0
$-
PSHELL* 2 1 2.0000003E-01 1*
* 1.0000000E+00 0 8.33333313E-01 0.0000000E+00*
* 0
$-
PBARL* 3 1 MSCBML0 HAT *
* 3.0000000E+00 1.31358892E-01 2.00000000E+00 8.99999976E-01*
* 0.0000000E+00
$-
MAT1 1 1.0E+7 0.33 0.283
$-
FORCE 1 10004 2000.0 1.0 0.0 0.0
$-
$ PARAM AUTOSPC YES
$-
$-----DESIGN MODEL:-----
$-
$...DEFINE THE DESIGN VARIABLES:
$-
$DESVAR ID LABEL XINIT XLB XUB DELXV
DESVAR * 1T-PLATE 1.12644322E-01 1.00000005E-03+D 1V
*D 1V 1.00000000E+01 1.00000000E+00
DESVAR * 2T-BOX 1.31358886E+00 1.00000005E-03+D 2V
*D 2V 1.00000000E+01 1.00000000E+00

```

Figure 5-2 Punch File Produced Using ECHO=PUNCH(NEWBULK)

```

$...RELATE THE DESIGN VARIABLES TO ANALYSIS MODEL PROPERTIES
$(LINEAR RELATIONS SO USE DVPRELL)
$...EXPRESS SHELL THICKNESSES AS FUNCTIONS OF X1 X2:
DVPRELL1 ID      TYPE    PID     FID     PMIN    PMAX   CO          +
$+ DVVIDD1 COEF1   DVID2   COEF2   ...           +DP1
DVPRELL1 1        PSHELL  1       T       0.01
+DP1   1          1.0
$...EXPRESS BOX THICKNESS AS A FUNCTION OF X2:
DVPRELL1 3        PBABL   3       DIM2
2          0.1
$.   EXPRESS BOX OFFSET LOCATIONS AS A FUNCTION OF PLATE THICKNESS AND
$   FIXED BOX DIMENSIONS
DVCREL1 10       CBAR    31      W3A          1.5
1          0.5
DVCREL1 20       CBAR    32      W3A          1.5
1          0.5
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 20      13      -0.1   0.1
DCONSTR 30      14      -0.03  0.03
$DCONADD DCID    DC1    DC2    ...
$ SUMMED CONSTRAINT SET FOR SUBCASE 1
DCONADD 100    10      20
$ SUMMED CONSTRAINT SET FOR SUBCASE 2
DCONADD 200    10      30
$...OPTIONAL OVERRIDE OF OPTIMIZATION PARAMETERS:
$DOPTPRM IPRINT 7      DESMAX 20      DELP    0.5    P1      1      +
+ P2      15
$ (DELP=0.5 ALLOWS LARGER MOVES  THUS OVERCOMING CONSTRAINT
$ VIOLATIONS QUICKER)
$ENDDATA

```

Figure 5-3 Punch File Produced Using ECHO=PUNCH(NEWBULK)



Postprocessing Output

This final section in the chapter discusses some of the postprocessing options that are available from SOL 200 analyses. In particular, the ability of Patran to postprocess optimization results is discussed. The immediate discussion is of a file that contains results data in a format that allows input into a spreadsheet, thereby providing access to the spreadsheet's plotting capability.

Comma Separated Values File

The [File Management](#) indicates how a file can be assigned for the special purpose of producing a file that can be read into a spreadsheet utility such as Microsoft Excel. These results can then be manipulated using the spreadsheet utilities and, in particular they can be plotted. The file is designated a Comma Separated Values (or CSV) because the data are delimited by commas so that the spreadsheet can readily understand them. In addition to the ASSIGN statement such as:

```
ASSIGN USERFILE='DSOUG4.CSV' , STATUS=NEW, FORM=FORMATTED, UNIT=52
```

it is necessary to include a PARAM, XYUNIT, 52 in the Bulk Data file. The DSAPRT command is required to produce formatted sensitivity data. Design cycle history data, including the objective, maximum constraint and design variable values are all available in this file. [Figure 5-4](#) shows four plots that can be derived from the CSV file that is produced when the DSOUG4 file of the [Stiffened Plate](#) is used. The first three plots show:

1. The sensitivity of the weight as a function of the two design variables
2. The sensitivity of the stress as a function of the two design variables
3. The ratio of the stress sensitivities to the weight sensitivities for the same two design variables.

This third chart is derived in Microsoft Excel by dividing the data of the second chart by that of the first. This shows that the most effective way to achieve the desired stress limit is to reduce the first design variable while increasing the second.

The fourth chart in [Figure 5-4](#) depicts the values of the design variables as a function of the design cycle. Similar plots could be produced for the objective and the maximum constraint value.



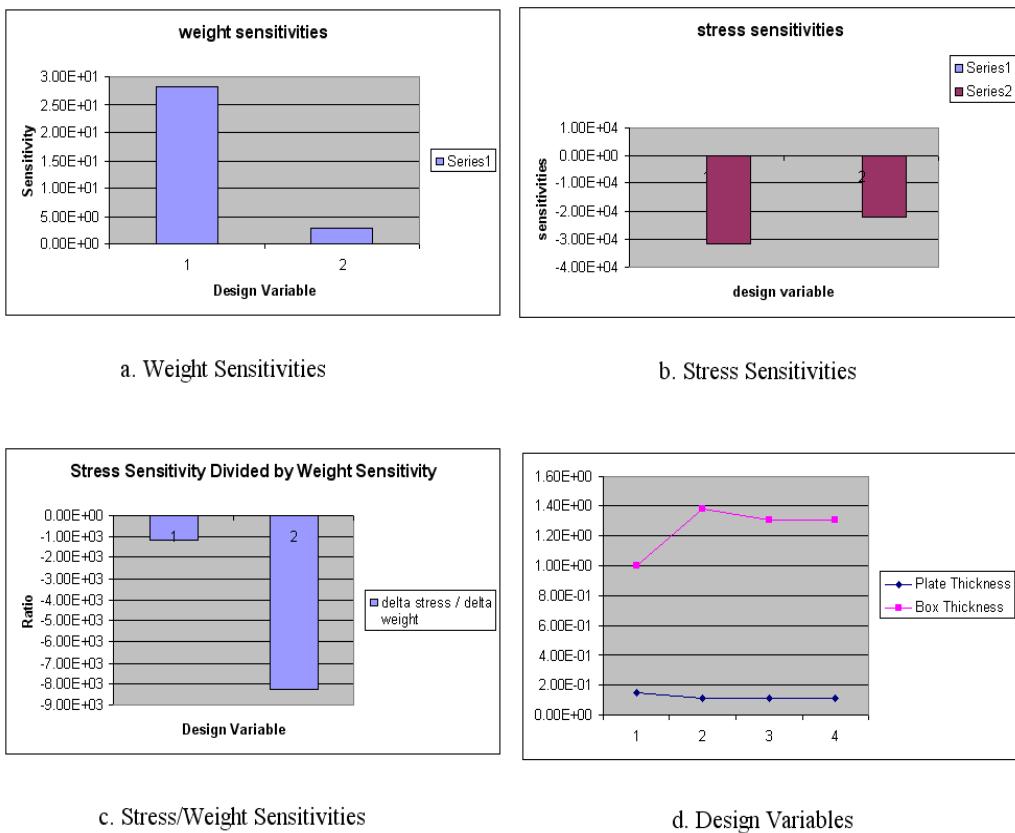


Figure 5-4 Plots Produced from a Comma Separated Values File

Output from PARAM POST

This section discusses the postprocessing support that is available in Patran for optimization results from MSC Nastran. There are two alternatives for creating output databases for Patran:

- PARAM POST 0 in the MSC Nastran bulk data file results in an .xdb file that can be attached in an Patran session.
- PARAM POST -1 results in an .op2 file that is read onto the Patran data base.

Both of these options support display of finite element results as a function of design cycle. (It is necessary to set PARAM NASPRT 1 to have data saved at each design cycle and only data requested in case control is saved.) Therefore, it is possible to graphically depict, for example, the stress patterns or the deformations in the designed structure at each design cycle.

The options differ in that the .op2 file contains some datablocks that are specific to SOL 200. These datablocks are documented in Section 4.4 of the *Patran MSC Nastran Preference Guide, Volume 1*:



Structural Analysis, which is available online when running Patran. In particular, a GEOMIN datablock permits the display of shape basis vectors and new shapes as a function of the design cycle. Another datablock named R1TABRG allows the display of which elements and grid responses are retained for the design task (see [Constraint Screening](#)). For element responses, elements with retained responses are identified in a on-off manner with retained elements marked in white while the non-retained elements are black. For grid responses, the direction of a marker at the grid point indicates which component of the grid response is retained.

The .op2 file also contains information on the design variables, the maximum constraint values and the objective function which are stored as global variables. These data are handled in a special way: when the .op2 file that contains these data is read into the Patran database, three XY plots are generated but not posted:

1. Objective function vs. design cycle
2. Maximum constraint vs. design cycle
3. Design Variable vs. design cycle.

[Figure 5-5](#) compares displacement results for the first subcase at the initial and final design cycles for the design task given in [Cantilevered Plate](#). Note the scales on the side of the two plots which indicates that the maximum displacement has increased from 0.218 to 2.00, which is equal to one of the design limits for this problem. [Figure 5-6](#) shows XY plots that are automatically generated in Patran that show the design objective and the three design variable values as a function of the design cycle.



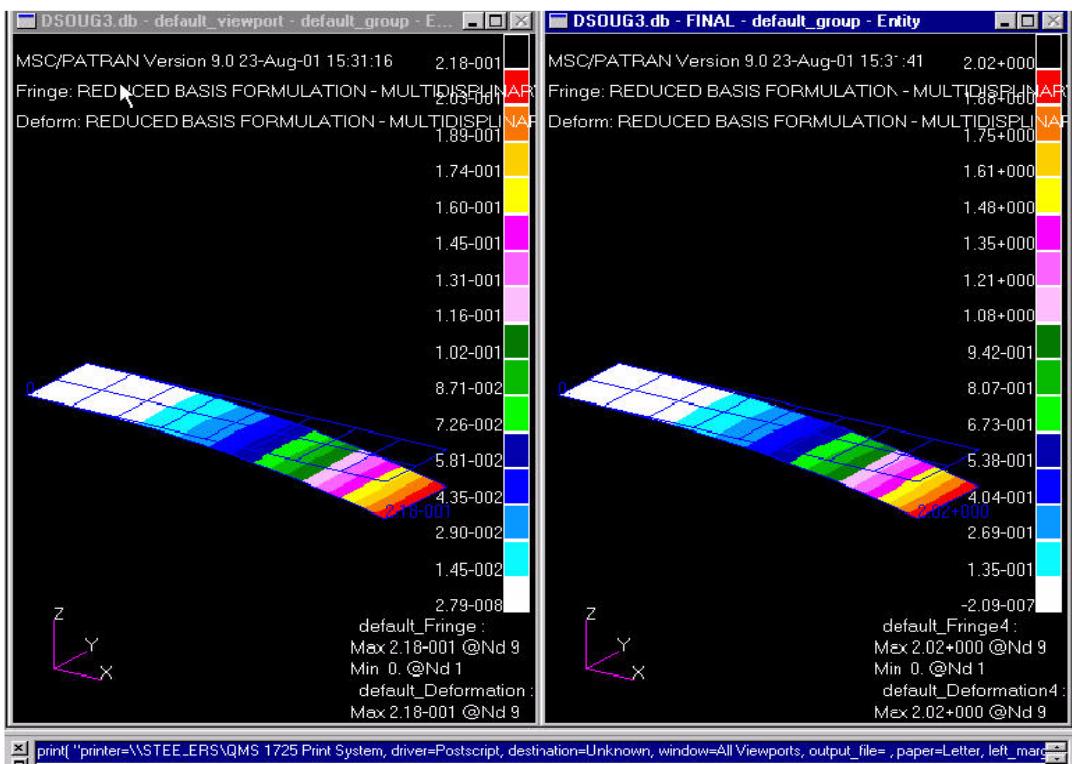
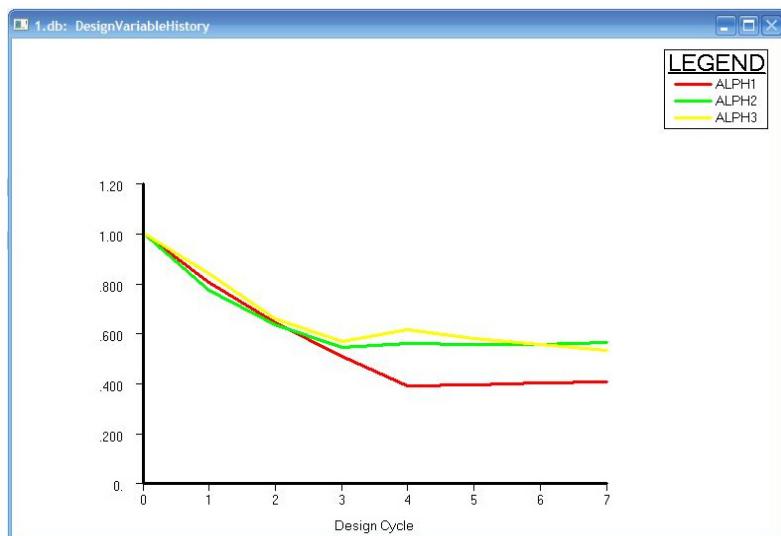
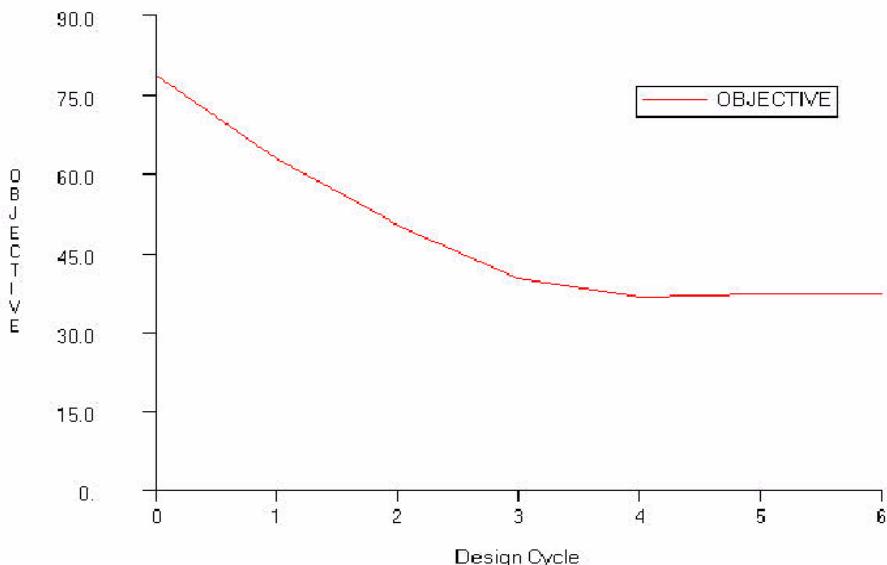


Figure 5-5 Displacement Patterns for the Initial and Final Design Object





a. Design variables as a function of design cycle



b. Design objective as a function of design cycle

Figure 5-6 Representative XY Plot from Patran





6

Shape Optimization

- Introduction 310
- Basis Vectors in Shape Optimization 311
- Relating Design Variables to Shape Changes 316
- Examples 343



Introduction

Shape sensitivity and optimization in MSC Nastran requires the definition of design variables, allowable shape variations or shape basis vectors and relationship between the design variables and shape basis vectors. A single shape or basis vector typically affects multiple grid locations. The amount the design variable is changed during optimization results in a corresponding shape change.

This chapter begins with a discussion of basis vectors. This is followed by a detailed description of tools available in MSC Nastran and Patran for relating design variables to the basis vectors. The final section illustrates these tools, explains the input deck requirements, and describes the special outputs and post processing support with example problems.



Basis Vectors in Shape Optimization

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

The angle bracket redesign of Figure 6-1 can be used to illustrate the concepts of shape basis vectors.

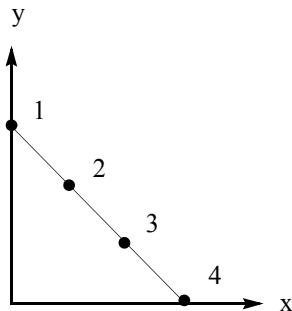


Figure 6-1 Angle Bracket

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 grid points 2 and 3 can easily be described in terms of the bracket leg lengths G_{1y} and G_{4x} . For simplicity, assuming only four grids along this edge, we can write

$$\begin{Bmatrix} G_{2x} \\ G_{2y} \\ G_{3x} \\ G_{3y} \end{Bmatrix} = \begin{bmatrix} 0 & 1/3 \\ 2/3 & 0 \\ 0 & 2/3 \\ 1/3 & 0 \end{bmatrix} \begin{Bmatrix} G_{1y} \\ G_{4x} \end{Bmatrix} \quad (6-1)$$

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 [Equation \(6-1\)](#), leaving its basic form unchanged.

We can write [Equation \(6-1\)](#) in terms of grid variations as

$$\begin{Bmatrix} \Delta G_{2x} \\ \Delta G_{2y} \\ \Delta G_{3x} \\ \Delta G_{3y} \end{Bmatrix} = \begin{bmatrix} 0 & 1/3 \\ 2/3 & 0 \\ 0 & 2/3 \\ 1/3 & 0 \end{bmatrix} \begin{Bmatrix} \Delta G_{1y} \\ \Delta G_{4x} \end{Bmatrix} \quad (6-2)$$

The coefficient matrix is unchanged in [Equation \(6-2\)](#) from that of [Equation \(6-1\)](#).

[Equation \(6-2\)](#) suggests that a convenient choice of design variables would be



$$\begin{aligned}\Delta x_1 &= \Delta G_{1y} \\ \Delta x_2 &= \Delta G_{4x}\end{aligned}\quad (6-3)$$

or

$$\begin{Bmatrix} \Delta G_{1y} \\ \Delta G_{2x} \\ \Delta G_{2y} \\ \Delta G_{3x} \\ \Delta G_{3y} \\ \Delta G_{4x} \end{Bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 1/3 \\ 2/3 & 0 \\ 0 & 2/3 \\ 1/3 & 0 \\ 0 & 1 \end{bmatrix} \begin{Bmatrix} \Delta x_1 \\ \Delta x_2 \end{Bmatrix} \quad (6-4)$$

Note that [Equation \(6-4\)](#) 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 [Equation \(6-4\)](#) 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.

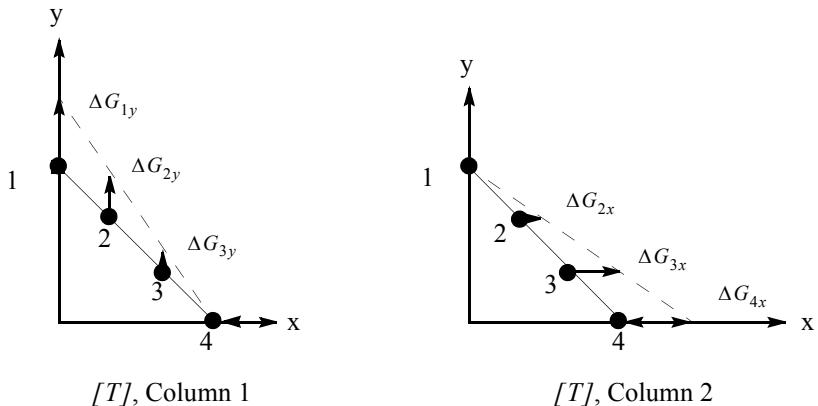


Figure 6-2 Basis Vectors for the Angle Bracket

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

We can generalize the relation given in [Equation \(6-4\)](#) as

$$\{\Delta G\}_{mx1} = [T]_{mxn} \{\Delta x\}_{nx1} \quad (6-5)$$



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

$$\{\Delta x\} = \{x\}^{i+1} - \{x\}^i$$

and i = design cycle number.

As the design variables change, the grid locations are computed as

$$\{G\}_{\text{new}} = \{G\}_o + [T]\{x_{\text{new}} - x_o\} \quad (6-6)$$

[Equation \(6-5\)](#) 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 $m \gg n$. For this reason, [Equation \(6-5\)](#) is often called a reduced basis formulation.

It needs to be reemphasized that the designed shape is the perturbation of the original shape defined by the GRID locations. The perturbation, as expressed in [Equation \(6-5\)](#), is the product of the shape basis vector and the **change** in the design variable value. This contrasts with sizing optimization, where the designed property is determined based on the specification of [Equation \(2-2\)](#) or [Equation \(2-5\)](#) and the input property from the analysis model is ignored.

Input of Shape Basis Vectors

MSC Nastran supports four methods to describe shape basis vectors:

1. Manual grid variation
2. Direct input of shapes
3. Geometric boundary shapes
4. Analytic boundary shapes

An overview of these methods is given here and details of the user interface in the next section, [Relating Design Variables to Shape Changes](#).

Manual Grid Variation

This is the most general, as well as the most tedious, method of generating shape basis vectors. 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 6-3](#) and [Equation \(6-7\)](#). A single DVGRID entry is required for every design variable-grid pair.



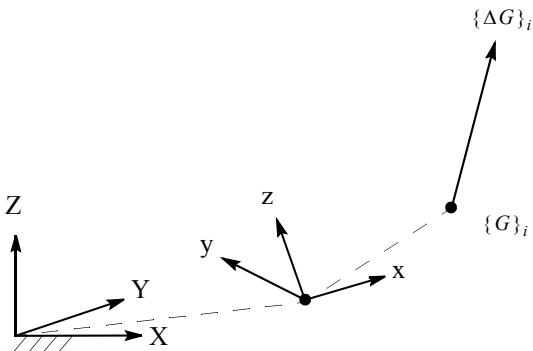


Figure 6-3 Grid Point Variation Defined by a DVGRID Entry

$$\begin{Bmatrix} \Delta G_x \\ \Delta G_y \\ \Delta G_z \end{Bmatrix}_i = \left(COEFF \begin{Bmatrix} N1 \\ N2 \\ N3 \end{Bmatrix} \right)_{ij} \Delta x_j \quad (6-7)$$

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. These externally generated vectors are solution vectors from an auxiliary model that shares the geometry (grid numbers and locations) with the primary designed model but the boundary conditions and analyses are chosen to produce deformations (or mode shapes) that are candidate shape changes. [Relating Design Variables to Shape Changes](#) details the steps required to use this method while [Shape Optimization of a Culvert](#) provides an example application of this technique.

Geometric Boundary Shapes

This method is similar to the Manual Grid Variation method except that the user is required to define the allowable shape variations only on the boundary of the structure. In addition to supplying the DVGRID information on the boundary, it is also necessary to supply BNDGRD Bulk Data entries to define the structure's boundaries. With this method, the shape basis vectors for the entire structure are automatically generated by the code through a process of interpolation of the boundary shape changes to the interior of the structure. An advantage of this method is that this interpolation is updated on every design cycle, minimizing the problems associated with mesh distortion for large shape changes.

Analytic Boundary Shapes

This approach extends the geometric boundary shapes method to avoid having to supply 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 refer to this model over the boundaries as an auxiliary boundary model. The designer then constrains and statically loads the auxiliary 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. Unlike the Direct Input of Shapes method, the auxiliary model is included in the same input file as the primary model and the analysis of the auxiliary model feeds the shape design task directly. [Relating Design Variables to Shape Changes](#) details the steps required to use this method while [Analytic Boundary Shape](#) provides an example application of this technique.



Relating Design Variables to Shape Changes

This section enlarges on this earlier discussion by identifying the specific bulk data entries that are used for each of these options. All of the methods, except for manual grid variation, use the concept of auxiliary model in some form. Therefore the auxiliary model concept is first discussed before proceeding to the details of the various methods.

Auxiliary Models in Shape Optimization

An auxiliary model is 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 the translational displacement components of these vectors as representing individual components of grid motion, we realize that the $\{U\}$ vectors can quite conveniently be used as basis vectors for shape optimization. For the “direct input of shapes” method, the auxiliary model is exercised in a separate job submittal and the results input into the shape optimization task. For the “geometric boundary shapes” and “analytic boundary shapes” methods, the auxiliary model is integrated with the shape optimization task so that the shape vectors are generated in the same job submittal as is used to perform the optimization task.

As a simple example, consider the cantilever structure of [Figure 6-4](#). 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 can use an auxiliary model to generate a few characteristic profiles and use these as our shape basis vectors.



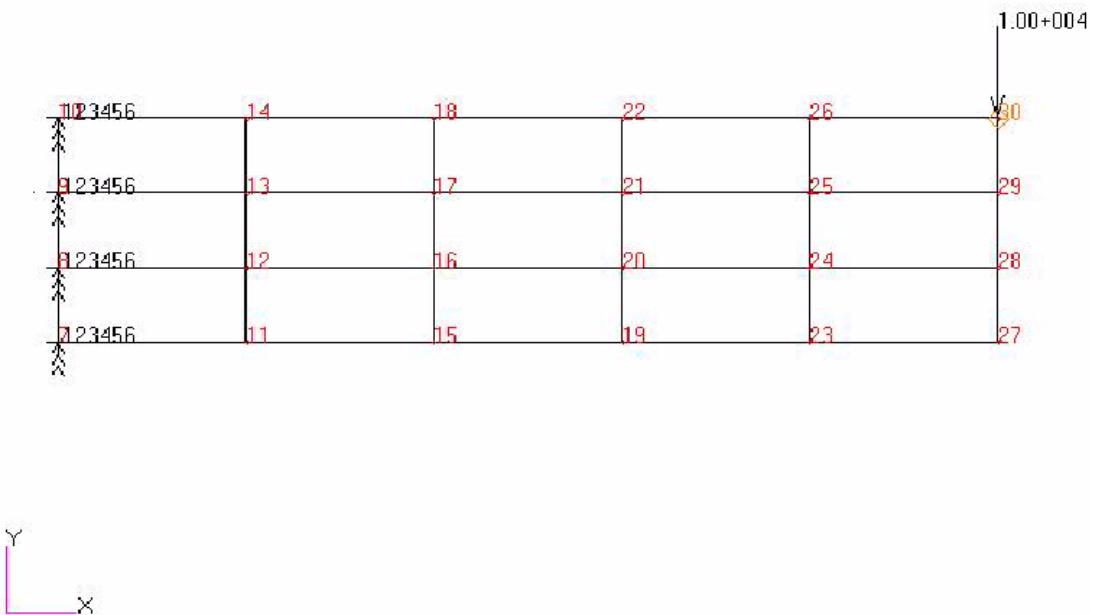


Figure 6-4 Simple 2-D Cantilever

Figure 6-5 shows the corresponding auxiliary model and its boundary conditions, exclusive of the loading. The fixed edges on the left and top sides guarantee that the left-hand support and the horizontal upper edge will not change during shape optimization. The support in the x direction along the right edge allows the depth of the beam to change without changing its length. We can now load the bottom edge in a way to produce a range of “shapes” or basis vectors.

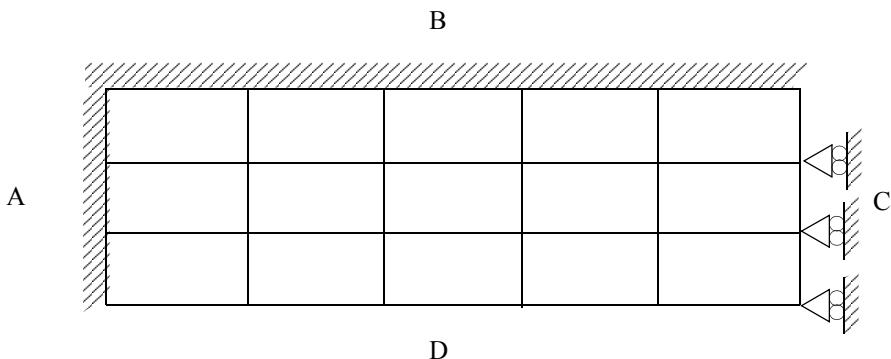


Figure 6-5 Auxiliary Model for the Simple 2-D Cantilever

To view the basis vector:



- a. Import the model,
- b. Make a load case (displacement and loads) to generate the kind of shape you want
- c. Select the subcases,
- d. Analyze and
- e. View the displacement vectors by subcase (which is the same as viewing the basis vectors, [Figure 6-6](#)).

This can be done for as many basis vectors as you want

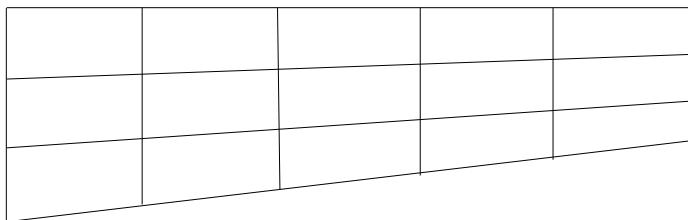


Figure 6-6 Uniform Taper Basis Vector

A more arbitrary shape basis vector could be generated by loading an individual grid along the lower edge. In order to avoid an inflection point in the resulting deflection pattern, it is advisable to add some very stiff beams to the bottom edge of the auxiliary model. This produces a shape as shown in [Figure 6-7](#). A number of these shapes produced by loading other grids along this edge might comprise a useful set of shape basis vectors.

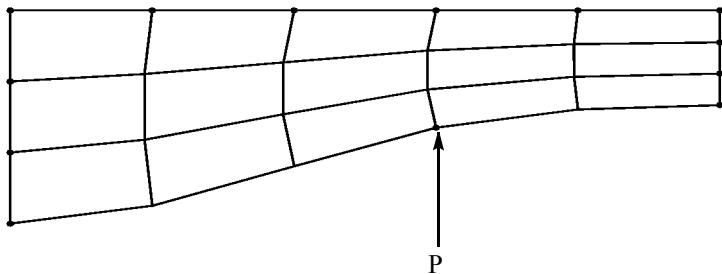


Figure 6-7 Point-Load Basis Vector

Modeling Methods (Shape Basis Vector Definition)

Shape optimization in MSC 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 MSC 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 MSC 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. Unfortunately, it cannot tell you how many holes to use or otherwise affect the topology of the model.

The purpose of each of the four methods of shape optimization is the same: to define a basis vector (or a set of basis vectors). The methods differ in terms of their user interfaces and whether or not the basis vectors are automatically updated for 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

In this method, DVGRID entries alone are used to describe shape changes. For the simple linear taper of [Figure 6-6](#), the DESVAR/DVGRID data is defined as follows:

DESVAR	10	LINTABP	1.0	0.01	10.		
DVGRID	10	11		1.0	0.0	0.4	0.0
DVGRID	10	12		1.0	0.0	0.2667	0.0
DVGRID	10	13		1.0	0.0	0.1333	0.0
DVGRID	10	15		1.0	0.0	0.8	0.0
DVGRID	10	16		1.0	0.0	0.5333	0.0
DVGRID	10	17		1.0	0.0	0.2667	0.0
DVGRID	10	19		1.0	0.0	1.2	0.0
DVGRID	10	20		1.0	0.0	0.8	0.0
DVGRID	10	21		1.0	0.0	0.4	0.0
DVGRID	10	23		1.0	0.0	1.6	0.0
DVGRID	10	24		1.0	0.0	1.067	0.0
DVGRID	10	25		1.0	0.0	0.5333	0.0
DVGRID	10	27		1.0	0.0	2.0	0.0
DVGRID	10	28		1.0	0.0	1.3333	0.0
DVGRID	10	29		1.0	0.0	0.6667	0.0

As mentioned, this represents a significant amount of data that must be generated in some fashion. The benefits, drawbacks and data preparation steps for this method are as follows:

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.



Data Preparation

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.
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. These vectors could be the results of a statics analysis, but other real vectors can also be used. For example, you may wish to use normal modes results. There are no DVGRID entries required in this method (although you can still use them to augment the direct input shapes). Instead, the DVSHAP entry is used to relate the DESVAR to the columns of the results from the auxiliary model analysis. The benefits, drawbacks and data preparation steps for this method are:

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, with results 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.

Data Preparation

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 MSC 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 DBLOCATEDd 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

In this method, BNDGRD Bulk Data entries are used to define the structure's boundaries and DVGRID entries to furnish the shape variations only over these boundaries. The linear taper basis vector of [Figure 6-6](#) can therefore be input using:

DESVAR	10	LINTAP	1.0	0.01	10.		
DVGRID	10	11		1.0	0.0	0.4	0.0
DVGRID	10	15		1.0	0.0	0.8	0.0
DVGRID	10	19		1.0	0.0	1.2	0.0
DVGRID	10	23		1.0	0.0	1.6	0.0
DVGRID	10	27		1.0	0.0	2.0	0.0
BNDGRID	123	11	15	19	23	27	7 8
	9	10	14	18	22	26	30
BNDGRID	1	28	29	30			

Note: It is necessary to specify the grid component combinations that are fixed using the BNGRID entry, but it is not necessary to provide a DVGRID entry when the grids are fixed to zero.

It is seen that the input preparation is simplified relative to the manual grid variation input for the same example. The benefits, drawbacks and data preparation for this method are:

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 significant DVGRID information, even though orders of magnitude less than the manual grid variation method.



Data Preparation

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 Shape

The analytic boundary shapes method can be characterized as being the most sophisticated of the methods and removes the need for DVGRID generation completely. Instead, you must generate an auxiliary model that "clads" the boundary that is to be redesigned. This can be done in a preprocessor, thereby minimizing the manual data preparation. The geometry (that is, GRID locations) is shared between the primary and auxiliary models and it is necessary to include BNGRID entries in the primary model that identify the boundary in the same way as in the geometry boundary shapes method. [Analytic Boundary Shapes](#) provides an example that uses this method and you should refer to this to get a clear picture of the requirements for generating the auxiliary models. This method involves separate bulk data for the auxiliary models, with each model denoted by a BEGIN BULK AUXMODEL = n command. Unique loads and boundary conditions are applied to the auxiliary model(s), necessitating separate case control sections for each of the models. The AUXCASE and AUXMODEL Case Control commands are used to select the separate bulk data files.

Benefits

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

Any combination of *static* loading available in MSC Nastran may be used. Point loads (FORCE), enforced displacements (SPCD), thermal loads (TEMP), pressure loads (PLOAD), and so on, may all be used to generate the desired shapes.

Drawbacks

Creativity is often required in the selection of loads and boundary conditions for the auxiliary boundary models. This is listed as a drawback because the "creativity" requires that you possess a certain amount of sophistication in model preparation. You may see this as a benefit in that your creativity can be applied to create the most appropriate shape basis vectors!

Data Preparation

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.



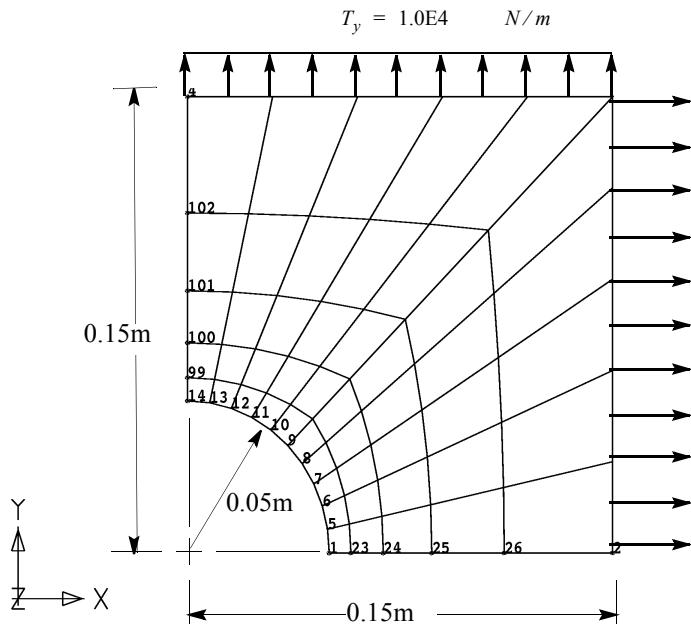
3. Define the shape boundary conditions on the actual structure using BNDGRID Bulk Data entries.
In addition to defining the connection points with the auxiliary boundary model, the BNDGRID entries must define those boundaries that are invariant during shape optimization.
4. Define the shape design variables 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 further 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 is 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 6-8](#). 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

$$E = 7.2\text{E}10 \quad \text{N/m}^2$$

$$\mu = 0.33$$

$$\rho = 2.8\text{E}3 \quad \text{kg/m}^3$$

Figure 6-8 Plate Quarter Model

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

$$\frac{x^2}{a^2} + \frac{y^2}{b^2} = 1 \quad (6-8)$$

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.

Note: From [Equation \(6-5\)](#) 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 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 MSC Nastran run.

[Figure 6-9](#) 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.

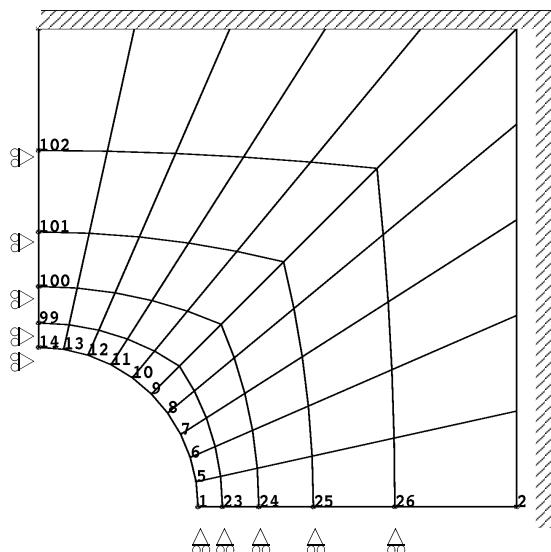


Figure 6-9 External Auxiliary Model

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, 101, 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, 104, 105
$
```

Note: The SPCD enforced displacement values correspond to an elliptical variation along the hole boundary with a maximum value of 0.01. The result of these load applications are shown in [Figure 6-10](#) and [Figure 6-11](#) (the deformations have been enlarged for clarity). These displacement vectors are used next to generate our shape basis vectors.



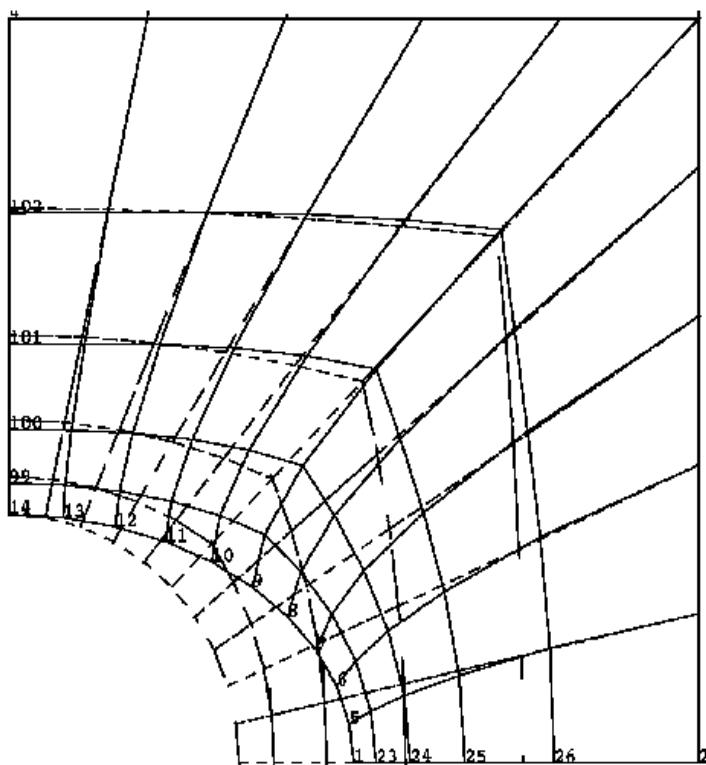


Figure 6-10 Displacement Vector 1



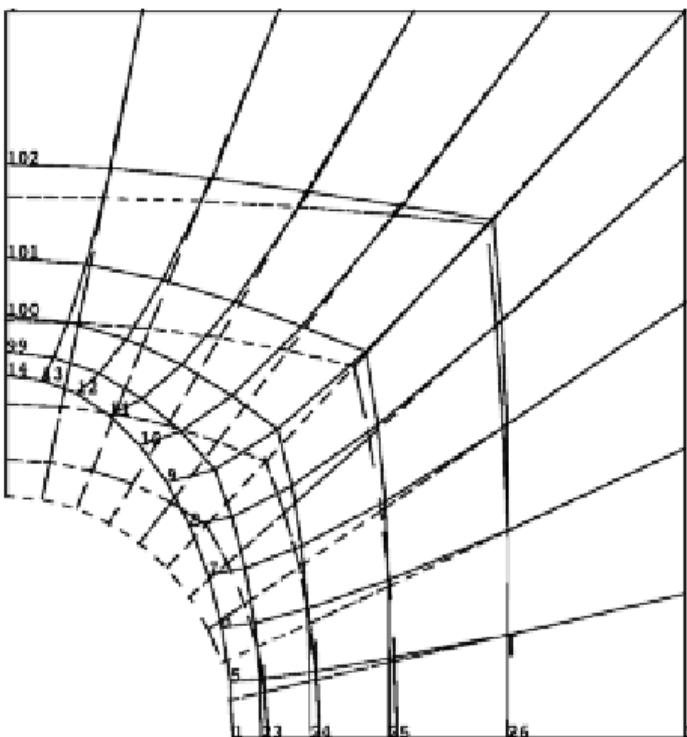


Figure 6-11 Displacement Vector 2

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=(ug7ugd,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 6-10](#) and [Figure 6-11](#).

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 x_1 and x_2 , the grid locations are varied according to

$$\{\Delta G\} = \begin{bmatrix} 1.0u_1 & 1.0u_2 \end{bmatrix} \begin{Bmatrix} \Delta x_1 \\ \Delta x_2 \end{Bmatrix} \quad (6-9)$$

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, with this method 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:

1. Grid components that change according to data supplied on the DVGRID entries.
2. 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
$ Y=0 PLANE...
BNDGRID,23456, 23,      24,      25,      26
$ X=0 PLANE...
BNDGRID,13456, 99,      100,     101,     102
$ CUTOUT EDGE...
BNDGRID,123456, 1,      5,       6,       7,       8,       9,       10,      +
+,      11,      12,      13,      14

```

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.

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.

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 6-10](#) and [Figure 6-11](#).

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 usually defined over the boundaries of the structure, although they can cover the whole structure if required. 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 for our plate example. This section must appear after the Bulk Data for the primary model.



```

BEGIN BULK AUXMODEL = 1
$ PARAM, PRGPST, NO
$ auxillary 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 as 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 will result in an elliptical-type variation in the x-direction, and the other 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 6-12](#) and [Figure 6-13](#).

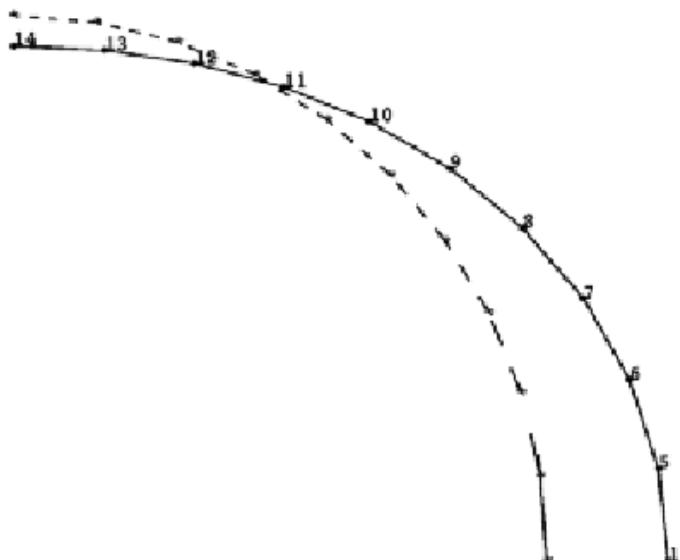


Figure 6-12 Boundary Displacement Set 1



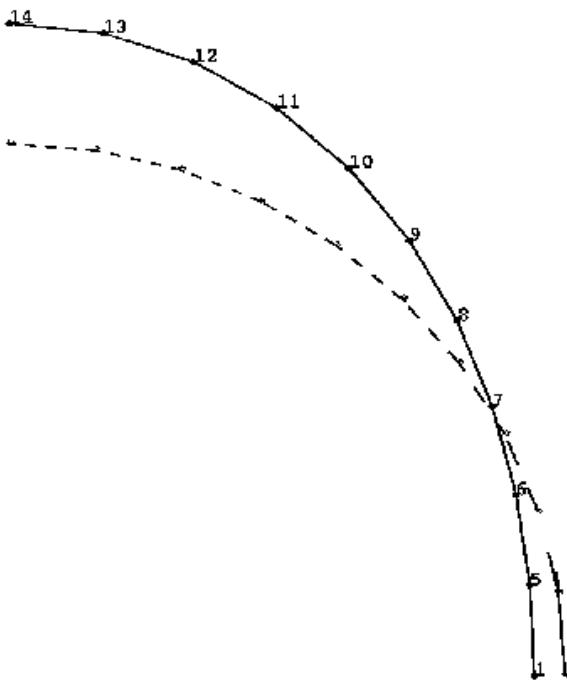


Figure 6-13 Boundary Displacement Set 2

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 being the same set of boundary conditions as 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:

$$\{\Delta G\} = \begin{bmatrix} 1.0 U_{AUXMID = 1} & 1.0 U_{AUXMID = 1} \\ SUBCASE = 10 & SUBCASE = 20 \end{bmatrix} \begin{Bmatrix} \Delta x_1 \\ \Delta x_2 \end{Bmatrix} \quad (6-10)$$

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.

Use of Patran to generate a Shape Optimization deck

Patran has a Shape Optimization Utility which may be used for generating design variables, basis vectors and DVGRID entries relating the basis vectors to the design variables. This feature is illustrated with the help of an example.

Model 1 - Plate with hole

This model is of a square plate with a reinforced hole in the middle. We want to vary the radius of the hole and of the stiffened section. Also the thickness of these plates can also be varied. The objective is to minimize mass and there are constraints on stress.



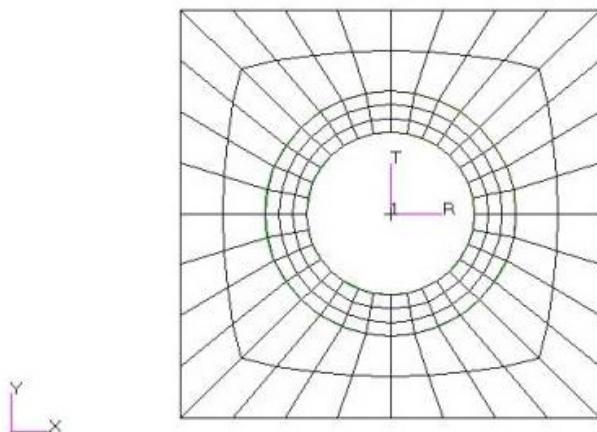


Figure 6-14 Plate with Hole

The plate is 10 units square. The starting hole diameter is 4 units with a stiffened area out to 6 units diameter.

Procedure

1. Create the model in Patran and create model variables for both pshell thicknesses using the standard Patran tool.
 - a. Go to Tools -> Design Study -> Pre Process



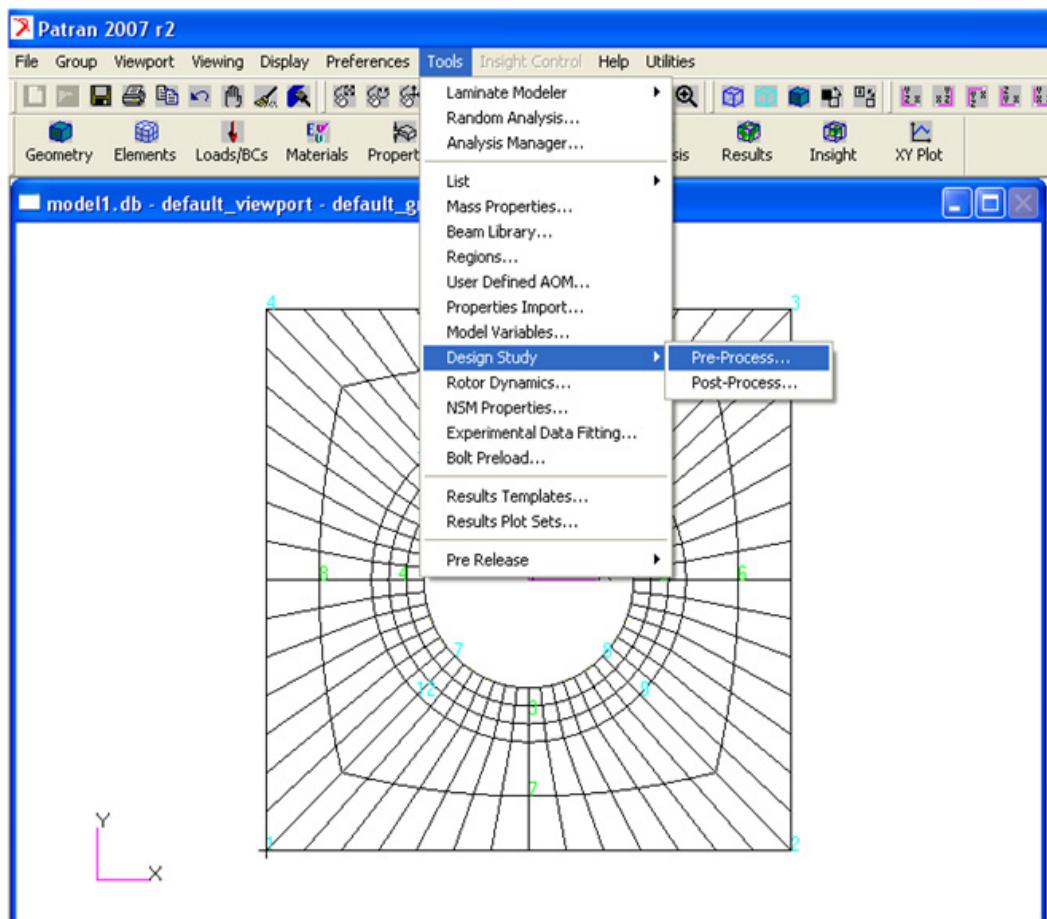


Figure 6-15 Pre-process in Patran

- b. Select Action-> Create, Object-> Design Variable, Type-> Property. Select Dimension->2D and Type->Shell. Select desired property from Select Property Set. Set Input Bounds at Lower/Upper and Value. Fill Variable Name, Analysis Model Value, and Lower (L) and Upper (U) bounds for the designed thickness.



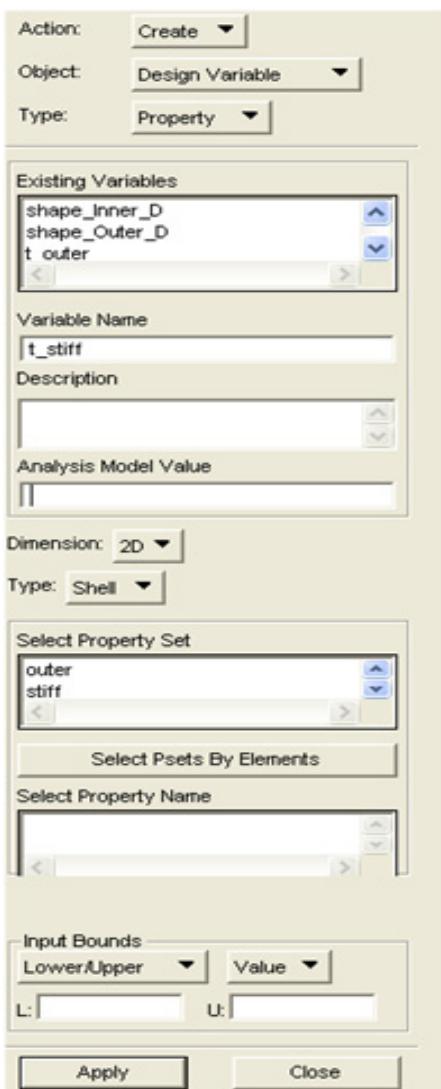


Figure 6-16 Creating Design Variables

2. Create fields describing the changes in radius. These fields should reduce in magnitude to zero at a radius away from the hole. (creating a zone of influence outside of which grid points are unaffected) Use Field names Inner_D and Outer_D. Each field is a vector PCL field defined using the cylindrical coordinate system 1. These fields must be of type vector to tell MSC Nastran which direction to move the grid points. Only the R component is defined in the field.
 - a. $\text{Inner_D} = ((4 - \text{R}) + \text{abs}(4 - \text{R})) / 4$
 - b. $\text{Outer_D} = ((4.5 - \text{R}) + \text{abs}(4.5 - \text{R})) / 4$



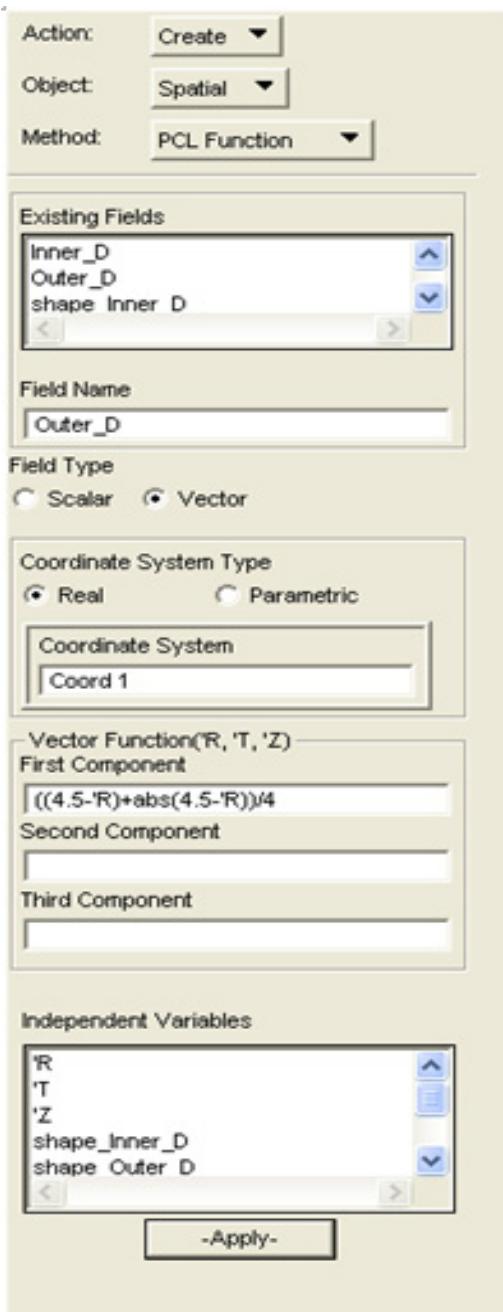


Figure 6-17 Creating Fields

3. Open the Shape utility form. Click Utilities->Analysis->Calculate Shape Basis Vectors



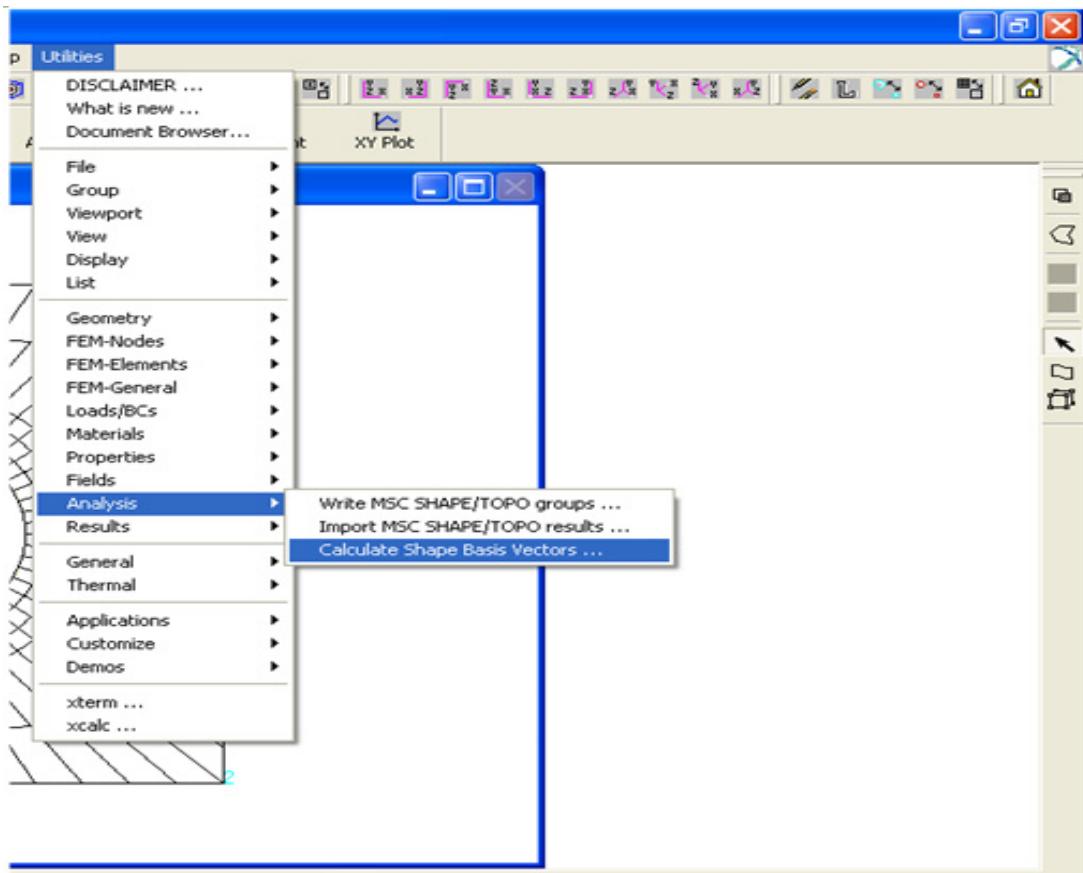


Figure 6-18 Shape utility form

- a. Select both fields
- b. Select all grid points other than those on the boundary and add to the select box
- c. Select "normalize maximum vector" and put 0.5 in the normalized value box to allow 0.5" maximum grid movement.
- d. Select "offset Desvar ID number" and put 2 in the box
- e. Select "apply"



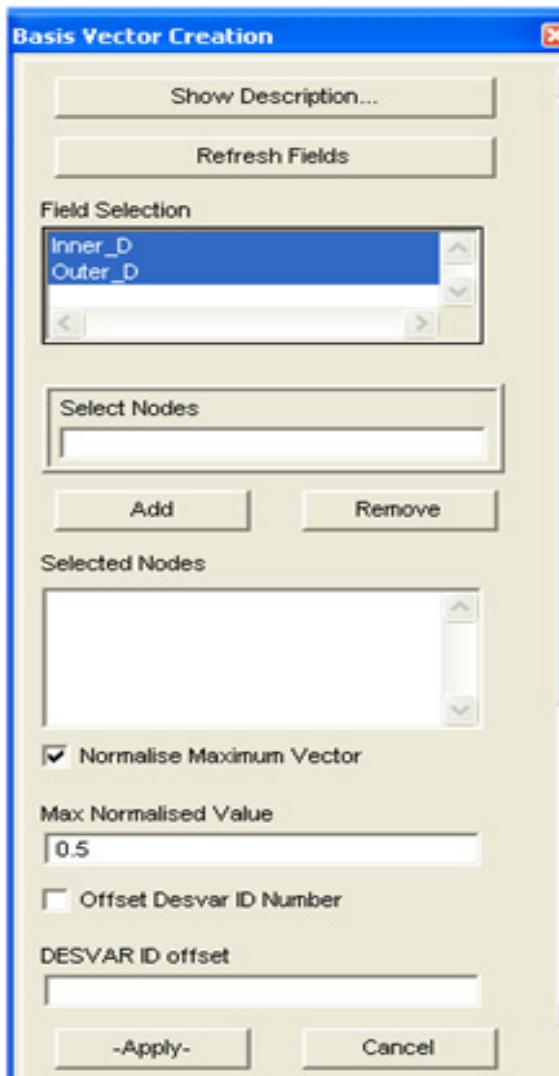


Figure 6-19 Basis vector creation

This will create a file called dbname.bv.nn where dbname is the name of your Patran database and nn is a unique number. The file includes all the relevant desvar and dvgrid entries for the shape optimization. There is also a line added in the direct text input for the current job:

```
include 'dbname.bv.nn'
```

This effectively inserts the basis vector into the analysis.



Examples

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 6-20 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 “Shape Optimal Design Using Isoparametric Elements” (see Reference 5.).

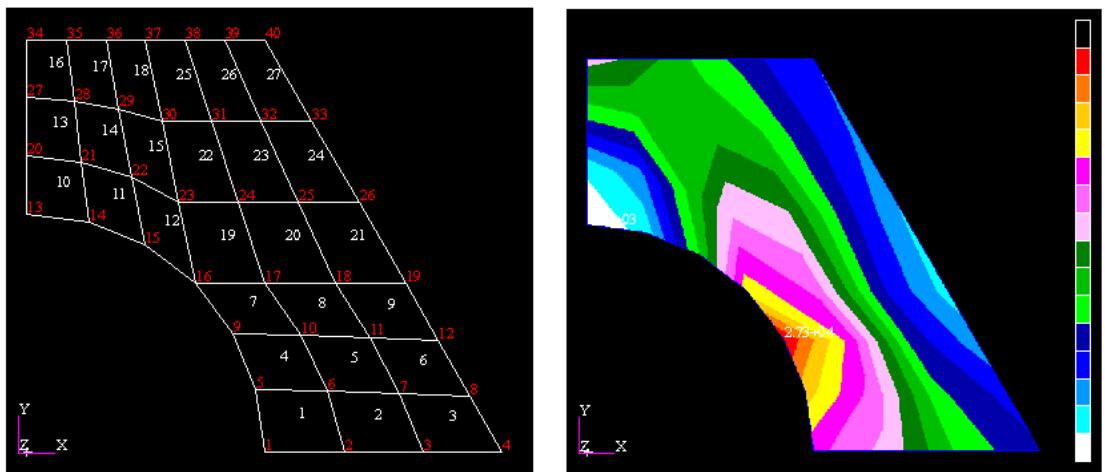


Figure 6-20 Initial Culvert Design

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 6-20](#). 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 6-21](#) shows the auxiliary model geometry. Note that it has the same geometry as the primary structure ([Figure 6-20](#)), but with different boundary and loading conditions. The input data file for this model is given in [Listing 6-1](#).

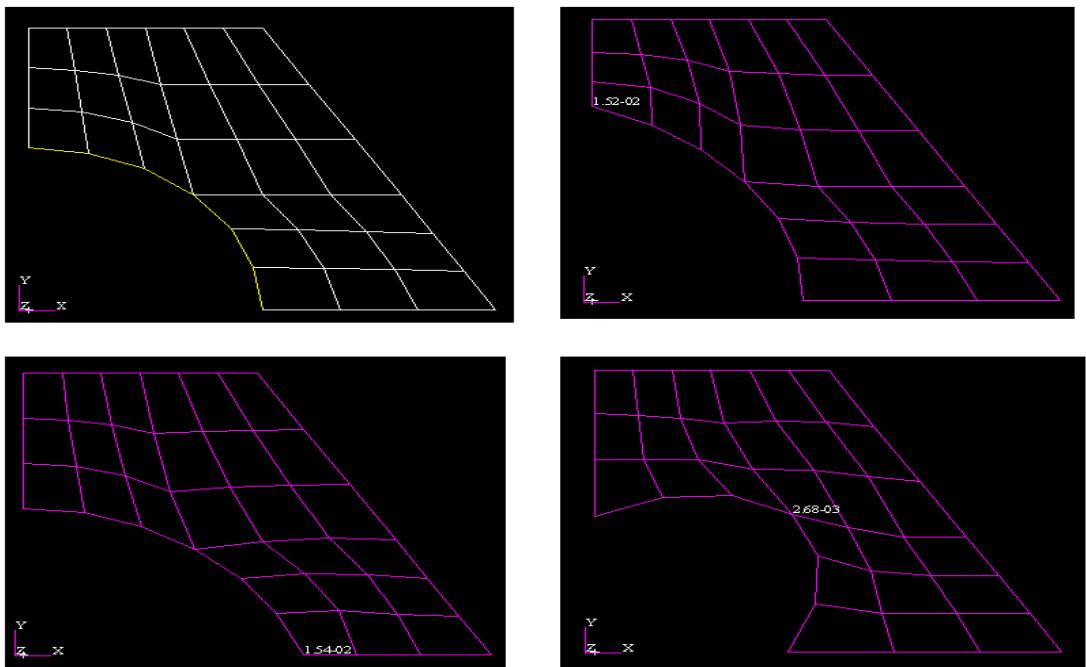


Figure 6-21 Auxiliary Model

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 6-21](#). Note from [Listing 6-1](#) 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 6-21](#), the first with a load applied at grid 13, the second at grid 16, and the third at grid 1.

To view shape basis vectors, import the model and results of external auxiliary model and click Result to obtain the Results form and select the options as shown in [Figure 6-22](#).

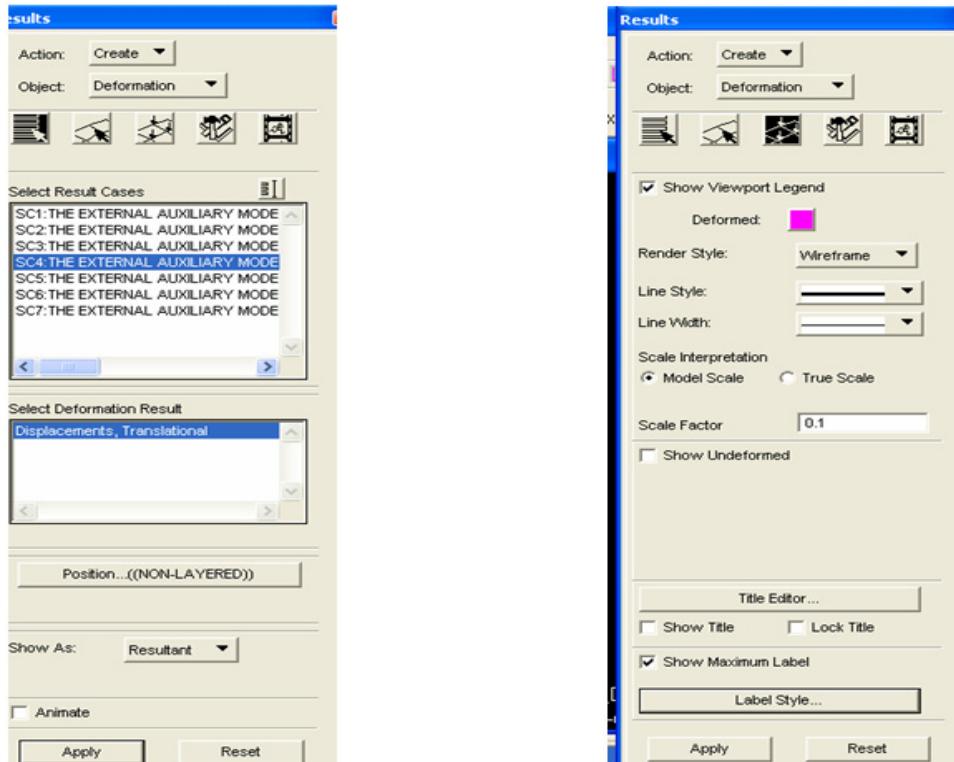


Figure 6-22 Visualizing Shape Basis Vector

The database is saved through the use of the SCR=NO or SCR=MINI keyword when submitting the auxiliary model job.

Listing 6-1 Auxiliary Model for the DSOUG5 Example (File DSOUG5A)

```
ID MSC, DSOUG5A $  
$ This is the external auxiliary model file for the shape  
$ optimization of a culvert example. This deck must be  
$ run first and the results saved to the database before the  
$ culvert example, dsoug5.dat, is run.  
TIME 10  
SOL 101  
CEND  
TITLE=Culvert Example Using External Auxiliary Model
```

DSOUG5



```

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 files)
.....
.....
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 files)
.....
.....
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

```

Design Optimization Input

In the optimization run, the displacement matrix containing the seven displacement vectors from the Auxiliary Model analysis are retrieved using the DBLOCATE statement. The optimization input file is shown in [Listing 6-2](#).

Listing 6-2 Input File for DSOUG5

```

$ DSOUG5.DAT
$ This is the input deck for the shape optimization of a culvert
$ example. It DBLOCATE's data from a prior auxiliary model run,
$ file DSOUG5A.DAT. That job must be run first and the results
$ saved to the database before this one is run.
$
$ FMS section for retrieving the auxiliary displacement matrix
$ For Linux/Unix platforms, the MASTER file name must be lower case.
$
ASSIGN F1_AUX='dsoug5a.MASTER'
dblocate datablk=(ug/ugd,geom1/geom1d,geom2/geom2d) ,
logical=f1 aux
ID MSC, DSOUG5 $
SOL    200   $
TIME   100
CEND
TITLE=CULVERT EXAMPLE USING EXTERNAL AUXILIARY STRUCTURE          DSOUG5
SUBTITLE=THE PRIMARY STRUCTURE
ANALYSIS = STATICS
SPC=25
LOAD=1
DISP=ALL
STRESS=all
DESSUB = 10
desobj = 5
BEGIN BULK
PARAM,POST,-1
param,cdif,no
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     .25
desvar  2      b2      3.      -1.e6    1.e6     .25
desvar  3      b3      3.      -1.e6    1.e6     .25
desvar  4      b4      3.      -1.e6    1.e6     .25
desvar  5      b5      3.      -1.e6    1.e6     .25
desvar  6      b6      3.      -1.e6    1.e6     .25
desvar  7      b7      3.      -1.e6    1.e6     .25
$
$ A DVSHAP entry defines a shape basis vector by associating one design
$ variable to a dblocated displacement.
$
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          3.100e4
doptprm DESMAX 25      APRCOD 1
param,nasprt,1
ENDDATA

```

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.

[Figure 6-24](#) shows a representation of the shapes at the end of design cycles 1, 2, 3 and 5 represented as deformation relative to the original grid.

To post process the shape change at the end of a chosen design cycle, import the model and results of the primary model and click Result to obtain the Results form and select the options as shown in [Figure 6-23](#).



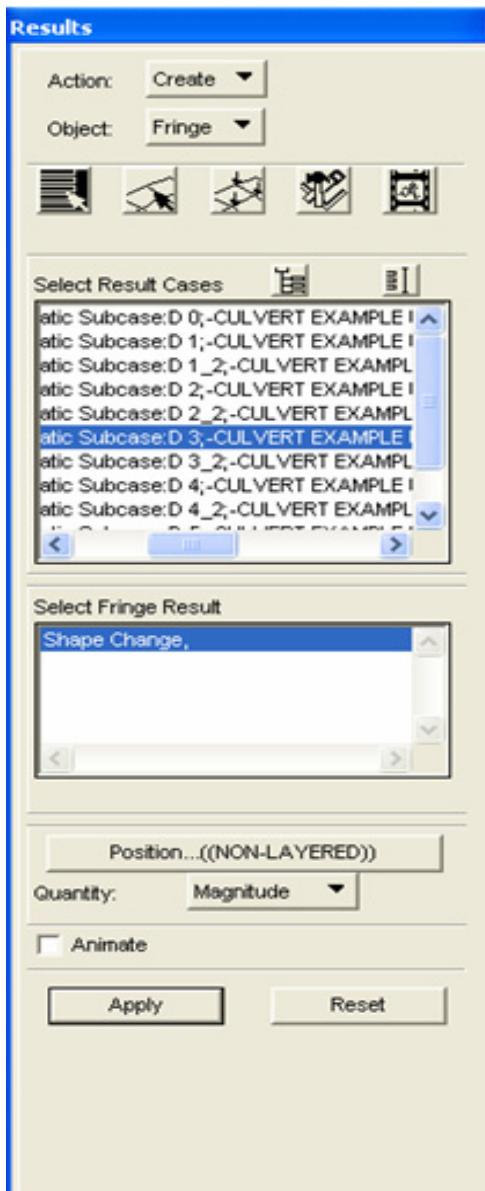
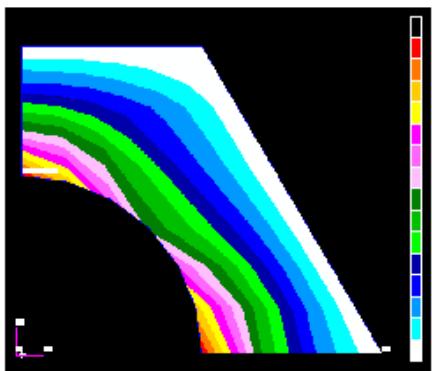


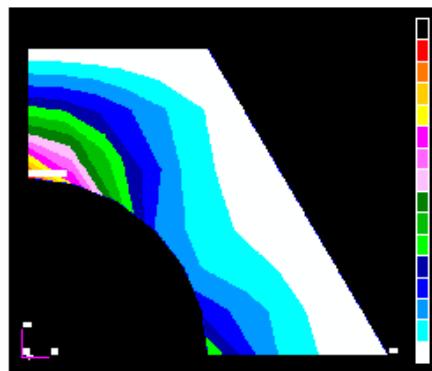
Figure 6-23 Visualizing Shape Change after each Design Cycle

Viewing these in Patran can provide guidance in the qualitative behavior of the shape changes as a function of design cycle.

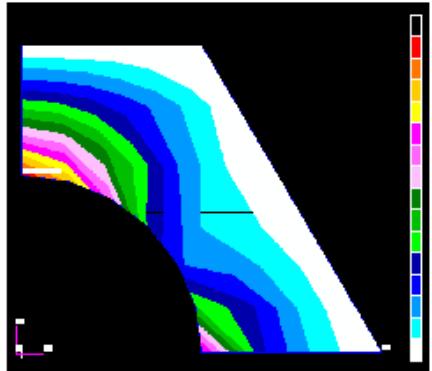




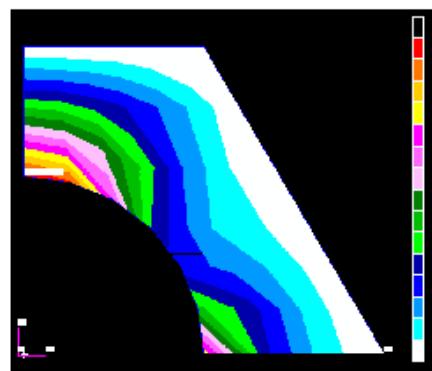
Design Cycle 15



Design Cycle 25



Design Cycle 35



Design Cycle 55

Figure 6-24 Designed Shapes for the Culvert as a Function of Design Cycle

[Listing 6-3](#) shows that the optimization task was completed in five design cycles and that the volume was decreased from 7.215 to 5.853, a decrease of 18.9% in volume (and weight) while still satisfying the stress limits.



Listing 6-3 Design History Output for DSOUG5

```

0
*****
SUMMARY OF DESIGN CYCLE HISTORY
*****
(HARD CONVERGENCE ACHIEVED)

NUMBER OF FINITE ELEMENT ANALYSES COMPLETED      6
NUMBER OF OPTIMIZATIONS W.R.T. APPROXIMATE MODELS   5

OBJECTIVE AND MAXIMUM CONSTRAINT HISTORY
-----
| CYCLE NUMBER | OBJECTIVE FROM APPROXIMATE OPTIMIZATION | OBJECTIVE FROM EXACT ANALYSIS | FRACTIONAL ERROR OF APPROXIMATION | MAXIMUM VALUE OF CONSTRAINT |
|-----|
| INITIAL      | 7.214704E+00                            |                           |                           | -1.021607E-01                |
| 1             | 6.714619E+00                            | 6.694779E+00              | 2.963468E-03                | 1.423551E-02                |
| 2             | 6.156898E+00                            | 6.163190E+00              | -1.020956E-03               | -1.690115E-02               |
| 3             | 5.842054E+00                            | 5.839865E+00              | 3.747831E-04                | 1.827577E-02                |
| 4             | 5.857889E+00                            | 5.857942E+00              | -9.035413E-06               | 1.642956E-03                |
| 5             | 5.852680E+00                            | 5.852699E+00              | -3.258921E-06               | 2.008821E-03                |
|-----|
1 CULVERT EXAMPLE USING EXTERNAL AUXILIARY STRUCTURE      DSOUG5      SEPTEMBER 14, 2001 MSC NASTRAN 4 / 9/01
PAGE 95
THE PRIMARY STRUCTURE
0
DESIGN VARIABLE HISTORY
-----
INTERNAL|EXTERNAL | DV. ID. | DV. ID. | LABEL | INITIAL : 1 : 2 : 3 : 4 : 5 :
|-----|
1 | 1 | B1 | 3.0000E+00 : 3.7500E+00 : 4.6875E+00 : 4.5473E+00 : 3.7210E+00 : 2.7907E+00 :
2 | 2 | B2 | 3.0000E+00 : 3.7500E+00 : 4.6875E+00 : 4.8695E+00 : 6.0869E+00 : 7.3548E+00 :
3 | 3 | B3 | 3.0000E+00 : 3.7466E+00 : 4.6833E+00 : 5.4171E+00 : 4.7637E+00 : 4.2786E+00 :
4 | 4 | B4 | 3.0000E+00 : 2.9473E+00 : 2.8163E+00 : 3.0271E+00 : 3.0831E+00 : 3.0930E+00 :
5 | 5 | B5 | 3.0000E+00 : 3.7500E+00 : 4.2426E+00 : 3.5757E+00 : 3.6450E+00 : 3.8176E+00 :
6 | 6 | B6 | 3.0000E+00 : 3.7500E+00 : 4.6870E+00 : 5.3197E+00 : 5.2333E+00 : 4.9588E+00 :
7 | 7 | B7 | 3.0000E+00 : 3.7500E+00 : 4.6875E+00 : 5.8594E+00 : 5.7831E+00 : 5.9582E+00 :

*** USER INFORMATION MESSAGE 6464 (DOM12E)
RUN TERMINATED DUE TO HARD CONVERGENCE TO AN OPTIMUM AT CYCLE NUMBER = 5.

```

The final shape is plotted in the left half of [Figure 6-25](#) while the final stress contour is plotted in the right half of the same figure.



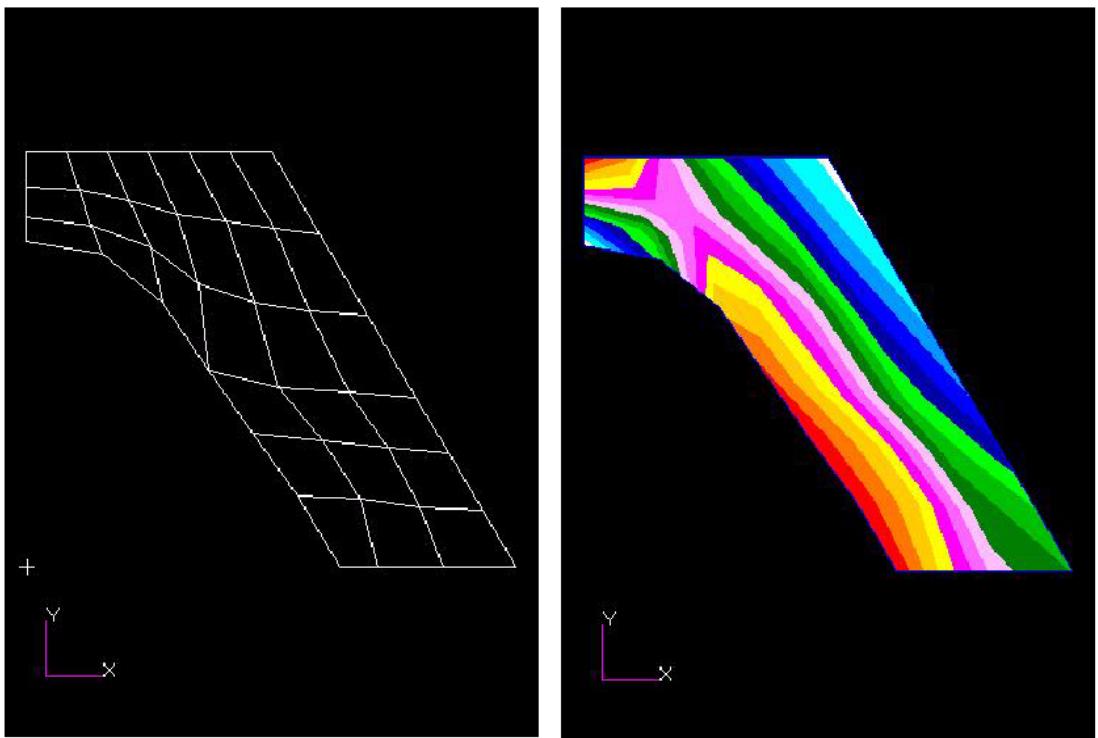


Figure 6-25 Final Culvert Design

Analytic Boundary Shapes

This example illustrates the use of the analytic boundary shapes method in shape optimal design. In this method, the entire modeling task can be written using the MSC Nastran input file alone, without the need for a modeling pre- and postprocessor. [Analytic Boundary Shapes](#) 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 6-26](#) 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 that the element von Mises stresses must be less than 200. We'll investigate minimizing the weight by tapering the cantilever's shape. The initial stress distribution is shown in



[Figure 6-27](#). The maximum stress is 183, and there are large regions where the stress is considerably less than the 200. limit.

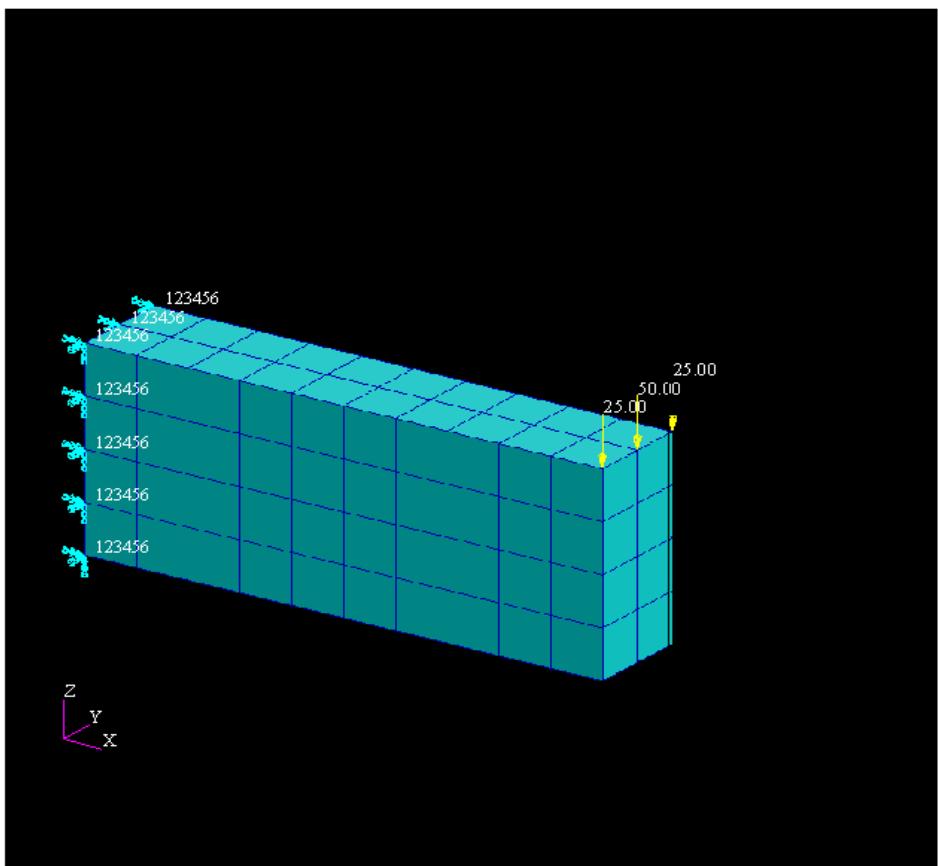


Figure 6-26 Solid Cantilever



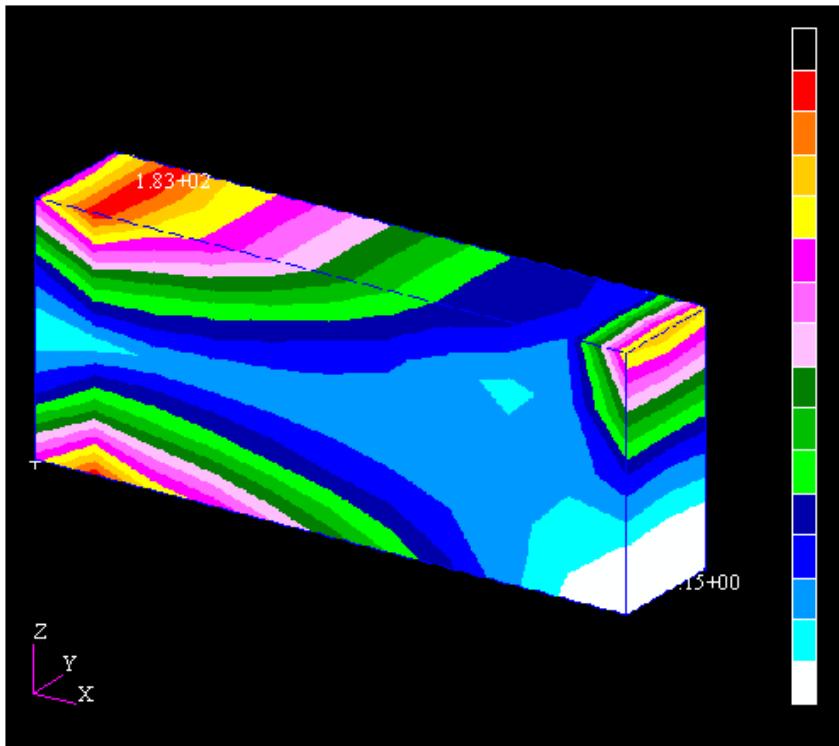


Figure 6-27 Solid Cantilever -- Initial Stress Distribution

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. In this

Boundary Shape Changes Using Auxiliary Boundary Models

[Figure 6-28](#) 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.



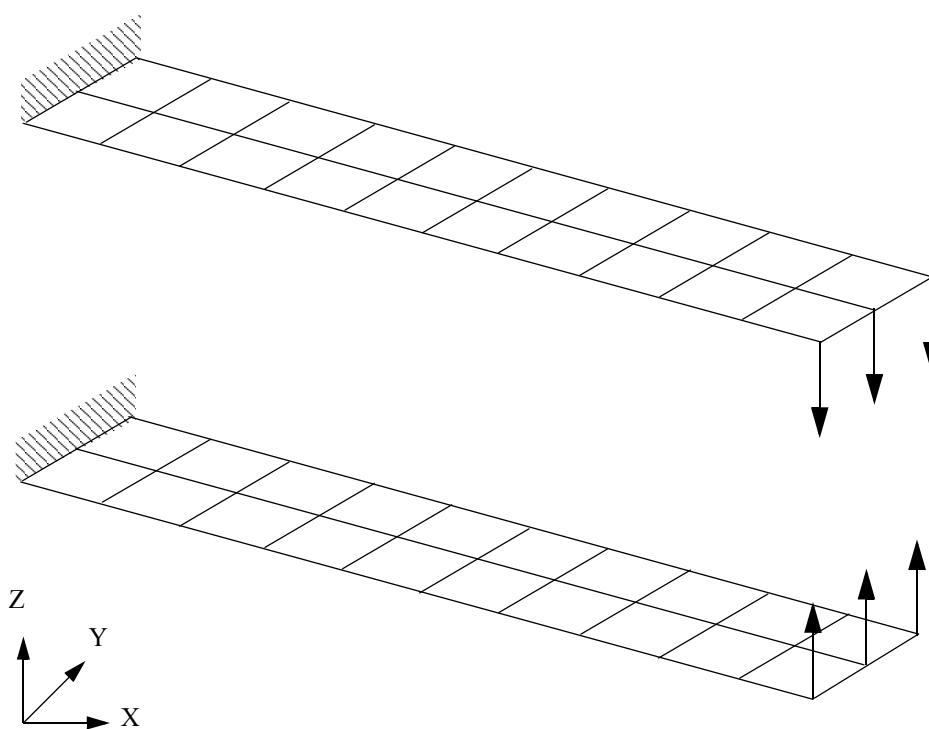


Figure 6-28 Auxiliary Boundary Models

In the example, a total of six boundary shapes are computed using the auxiliary model:

- Unit enforced displacements are imposed at the root and the tip of the top plate, resulting in a uniform variation along the top surface.
- A unit displacement is enforced at the tip of the top plate while the root is constrained in translation but allowed to rotate about the y-axis. This produces a shape that is linear along the top surface.
- An enforced displacement is applied at the tip while the root is fully constrained to produce a cubic displacement along the top surface. Note that a quadratic displacement could be produced by applying a moment at the tip of the plate.
- The remaining three shapes are produced by imposing similar conditions on the bottom surface while holding the top surface fixed.

[Listing 6-4](#) 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 DSOUG6.DAT. The primary model definition is for static analysis and is conventional, so its description is omitted here. The design model portion of the primary structure will be described shortly.



Listing 6-4 Selected Portions of Input File DSOUG6



```

GRID      164      0   9.000   2.000   4.000      0
GRID      165      0  10.000   2.000   4.000      0
$GRDSET
CHEXA     1       1       1       2      13      12      456
+EA       1       46      45
CHEXA     2       1       2       3      14      13      35
+EA       2       47      46
..... .
..... .
..... .
CHEXA     80       1    120     121     132     131     153     154+EA   80
+EA      80      165     164
MAT1*      1      2.0680E+05
*MA      1  1.00000000 1.169999996E-05
*MB      1 1500000.00 1500000.00      68000.00
*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,NASPRRT,1
$-
$DESVAR, ID,      LABEL,      XINIT,      XLB,      XUB,      DELXV
DESVAR     1  UPPERC      1.0      .00     7.00      0.1
DESVAR     2  LOWERC      1.0      .00     7.00      0.1
DESVAR     3  UPPERL      1.0      .00     7.00      0.1
DESVAR     4  LOWERL      1.0      .00     7.00      0.1
DESVAR     5  UPPERF      1.0      .00     7.00      0.1
DESVAR     6  LOWERF      1.0      .00     7.00      0.1
$-
$DVBSHAP,DVID,  AUXMID,  COL1,      SF1,      COL2,      SF2,      ...
DVBSHAP 1      1       1       1.0
DVBSHAP 2      1       2       1.0
DVBSHAP 3      1       3       1.0
DVBSHAP 4      1       4       1.0
DVBSHAP 5      1       5       1.0
DVBSHAP 6      1       6       1.0
$-
$DLINK, ID,      DDVID,      CO,      CMULT,      IDV1,      C1,      IDV2,      C2,      +
$+,      IDV3,      C3,      ...
$LINK 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     THRU     165
$
$ ---BOTTOM SURFACE:
BNDGRID 123     1     THRU     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     44     55     66     77     88     99     110
        121    132
$
$ ---FIXED END:
BNDGRID 1     34     45     56     67     78     89     100
        111    122
$
$ 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
DRESP1 3     STRESS  STRESS  PSOLID     34     1
DRESP1 4     STRESS  STRESS  PSOLID     55     1
DRESP1 5     STRESS  STRESS  PSOLID     76     1
DRESP1 6     STRESS  STRESS  PSOLID     97     1
DRESP1 7     STRESS  STRESS  PSOLID    118     1
DRESP1 8     STRESS  STRESS  PSOLID    139     1
DRESP1 9     STRESS  STRESS  PSOLID    160     1
DRESP1 10    STRESS  STRESS  PSOLID    181     1
DCONSTR 100   2     200.
DCONSTR 100   3     200.
DCONSTR 100   4     200.
DCONSTR 100   5     200.
DCONSTR 100   6     200.
DCONSTR 100   7     200.
DCONSTR 100   8     200.
DCONSTR 100   9     200.
DCONSTR 100   10    200.
$
$ OVERRIDE OF OPTIMIZATION PARAMETERS
DOPTPRM DESMAX 20     P1     1     P2     15METHOD3
$
$-----
$ 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
$ USE ENFORCED DISPLACEMENTS TO CREATE A UNIFORM TRANSLATION ON THE
$ LOWER SURFACE
0.0
SPC1        200     123     1     12     23
SPC1        200     123     11    22     33
SPC1        200   123456   34    THRU    165
SPCD        220     1     3     1.0    12      3     1.0
SPCD        220     23    3     1.0
SPCD        220     11    3     1.0    22      3     1.0
SPCD        220     33    3     1.0
$ USE ENFORCED DISPLACEMENTS TO CREATE A UNIFORM TRANSLATION ON THE
$ UPPER SURFACE
SPC1        300     123    133    144    155
SPC1        300     123    143    154    165
SPC1        300   123456   1    THRU    132
SPCD        330     133    3    -1.0    144      3     -1.0
SPCD        330     155    3    -1.0
SPCD        330     143    3    -1.0    154      3     -1.0
SPCD        330     165    3    -1.0
$ USE ENFORCED DISPLACEMENTS TO CREATE A LINEAR TRANSLATION ON THE
$ LOWER SURFACE (FREE UP THE 5 ROTATION AT THE ROOT)
SPC1        400   123456   1     12     23
SPC1        400     12    11     22     33
SPC1        400   123456   34    THRU    165
SPCD        440     11     3     1.0    22      3     1.0

```



```

SPCD    440      33      3      1.0
SPC1   400       3      11     22      33
$ USE ENFORCED DISPLACEMENTS TO CREATE A LINEAR TRANSLATION ON THE
$ UPPER SURFACE (FREE UP THE 5 ROTATION AT THE ROOT)
SPC1   500    12346   133    144    155
SPC1   500      12    143    154    165
SPC1   500    123456   1      THRU   132
SPCD   550      143      3    -1.0    154      3    -1.0
SPCD   550      165      3    -1.0
SPC1   500      3      143    154    165
$ USE ENFORCED DISPLACEMENT AT TIP AND RESTRAINED ROOT TO CREATE A
$ FLEXIBLE SHAPE ON THE LOWER SURFACE
SPC1   600    123456   1      12      23
SPC1   600      12      11     22      33
SPC1   600    123456   34      THRU   165
SPCD   660      11      3      1.0    22      3      1.0
SPCD   660      33      3      1.0
SPC1   600      3      11     22      33
$ USE ENFORCED DISPLACEMENT AT TIP AND RESTRAINED ROOT TO CREATE A
$ FLEXIBLE SHAPE ON THE UPPER SURFACE
SPC1   700    123456   133    144    155
SPC1   700      12    143    154    165
SPC1   700    123456   1      THRU   132
SPCD   770      143      3    -1.0    154      3    -1.0
SPCD   770      165      3    -1.0
SPC1   700      3      143    154    165
ENDDATA

```

Turning first to the auxiliary boundary model specification, we see that the boundary model is 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 MSC 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 six SPC sets that create the six boundary shapes identified above and use enforced motion (SPCD's) to supply the load. The first set of the SPC/SPCD conditions is explained here to help understand this concept. First the grids at the root and tip of the lower plate are constrained in the translational degrees of freedom as part of SPC SET 200. All of the remaining grids in the structure that are not on the lower plate are constrained using SPC SET 200 in all six degrees of freedom so that these grid locations will not move. SPCD SET 220 is then used to apply a unit enforced displacement at the root and the tip in the z-direction. Note that the points on the interior of the bottom plate are not constrained and are therefore free to move. In this case the whole plate will move as a rigid body in the z-direction, thereby providing the boundary shape. 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 six SPCD-defined loads, using a different boundary condition for each. Static loading is always assumed for these auxiliary model analyses.

The results of these six analyses are sets of six displacement vectors can be used to describe the 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. 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.

The BNDGRID data in this example are seen to be in three sets. The first set is applied to the grids along the top and bottom surface where it is seen that components 1,2 and 3 are identified, which means that these displacements will be determined from the auxiliary model. The second set is applied to grids that are on the side faces of the structure (excluding those at the top and bottom corners). These grids are restrained from moving in the y-direction so that the sides of the structure do not bulge out or contract. These side grids can move in the x and z directions. The third set of grids is made up of points at the root and tip of the structure. Here, the shape is not allowed to move in the x direction, while the x and z directions are left free.

Figure 6-29 shows the six deformations that result on the primary model when the displacement from the auxiliary model is applied.

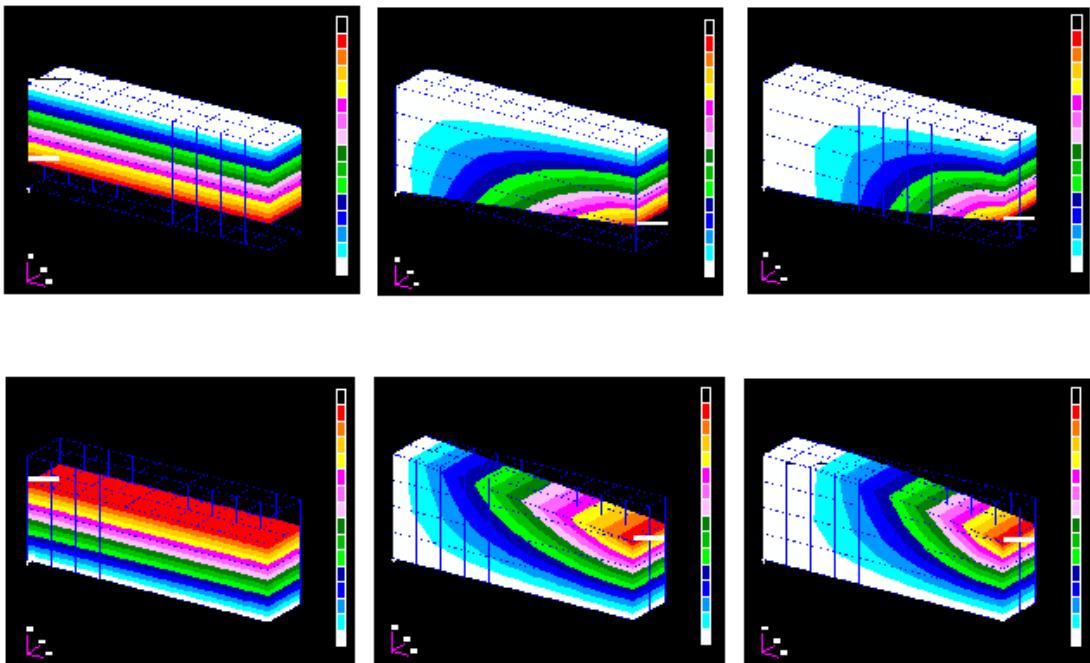


Figure 6-29 Shape Basis Vectors Shown as Deformations on the Primary Model



To obtain these shape basis vectors, run the deck with primary and auxiliary model with PARAM, OPTEXIT, 2 and import the model and results into Patran. Then click Results to get the Results form and select the options as shown in Figure 6-30.

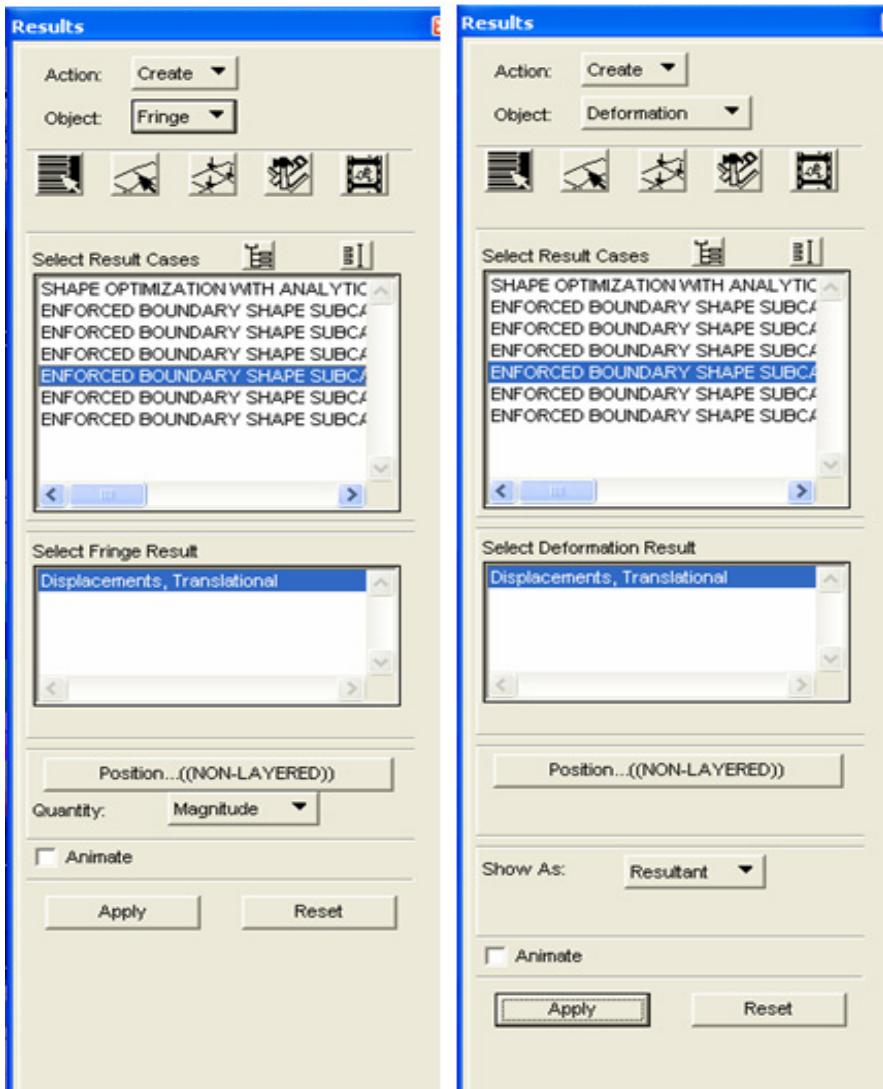


Figure 6-30 Visualizing Shape Basis Vectors from static analysis of Auxiliary models

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-labeled



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 through 6 relate design variables 1 through 6, respectively, to the six displacement solutions in a one-to-one fashion. Thus, each shape basis vector is simply 1.0 times each of the resultant displacement solutions.

Design Task

The design task is to modify the shape of the structure to find the minimum weight structure that satisfies the limit that the von Mises cannot exceed 200. It is seen that a DRESP2 is used to magnify the weight by 1.0E5. This DRESP2 is then selected as the objective. This step is not necessary in this case, but it does provide the optimizer with a sizable objective. Nine DRESP1 entries are used to identify von Mises stress responses at each corner, plus the center, of each CHEXA element. An upper bound limit of 200. is applied to these responses using DCONSTR entries. From the DOPTPRM entry, it is seen that the Sequential Quadratic Programming method of optimization is selected using METHOD=3.

Optimization Results

[Listing 6-5](#) shows the summary of design cycle history from the output file. In thirteen design cycles, the objective has been reduced from 8.0 E+6 to 6.01 E+6. The final shape is shown in [Figure 6-33](#), and the corresponding stress distribution in [Figure 6-34](#). It is seen that the final design is nearly symmetric and that much of the stress along the top and bottom surfaces is at the limit of 200.



Listing 6-5 Summary of Design Cycle History for DSOUG6

0

SUBCASE 100

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 13
NUMBER OF OPTIMIZATIONS W.R.T. APPROXIMATE MODELS 12

OBJECTIVE AND MAXIMUM CONSTRAINT HISTORY

CYCLE NUMBER	OBJECTIVE FROM APPROXIMATE OPTIMIZATION	OBJECTIVE FROM EXACT ANALYSIS	FRACTIONAL ERROR OF APPROXIMATION	MAXIMUM VALUE OF CONSTRAINT
INITIAL		8.000000E+06		-1.038307E-02
1	7.622278E+06	7.622278E+06	0.000000E+00	4.371643E-05
2	7.254322E+06	7.254322E+06	0.000000E+00	-1.355743E-04
3	6.847802E+06	6.847802E+06	0.000000E+00	-1.314545E-04
4	6.399672E+06	6.399672E+06	0.000000E+00	-1.700592E-04
5	6.285760E+06	6.285760E+06	0.000000E+00	1.011421E-02
6	6.212054E+06	6.212054E+06	0.000000E+00	1.214539E-02
7	6.171776E+06	6.171776E+06	0.000000E+00	8.254166E-03
8	6.119370E+06	6.119370E+06	0.000000E+00	8.320237E-03
9	6.082725E+06	6.082725E+06	0.000000E+00	6.020203E-03
10	6.039925E+06	6.039925E+06	0.000000E+00	6.038284E-03
11	6.008418E+06	6.008418E+06	0.000000E+00	4.509506E-03
12	6.006243E+06	6.006243E+06	0.000000E+00	-3.089905E-05

1 AUXILIARY MODEL 1

FEBRUARY 13, 2003 MSC.NASTRAN 2/12/03 PAGE 336

0

SUBCASE 100

DESIGN VARIABLE HISTORY

INTERNAL DV. ID.	EXTERNAL DV. ID.	LABEL	INITIAL	:	1	:	2	:	3	:	4	:	5	:
1	1	UPPERC	1.0000E+00	:	1.0070E+00	:	1.0028E+00	:	9.9867E-01	:	9.9434E-01	:	9.8921E-01	:
2	2	LOWERC	1.0000E+00	:	1.0070E+00	:	1.0028E+00	:	9.9867E-01	:	9.9434E-01	:	9.8921E-01	:
3	3	UPPERL	1.0000E+00	:	1.1000E+00	:	1.2100E+00	:	1.3310E+00	:	1.4641E+00	:	1.6105E+00	:
4	4	LOWERL	1.0000E+00	:	1.1000E+00	:	1.2100E+00	:	1.3310E+00	:	1.4641E+00	:	1.6105E+00	:
5	5	UPPERF	1.0000E+00	:	1.1000E+00	:	1.2100E+00	:	1.3310E+00	:	1.4641E+00	:	1.3989E+00	:
6	6	LOWERRF	1.0000E+00	:	1.1000E+00	:	1.2100E+00	:	1.3310E+00	:	1.4641E+00	:	1.3177E+00	:
INTERNAL DV. ID.	EXTERNAL DV. ID.	LABEL	6	:	7	:	8	:	9	:	10	:	11	:
1	1	UPPERC	9.8574E-01	:	9.8194E-01	:	9.7848E-01	:	9.7548E-01	:	9.7295E-01	:	9.7055E-01	:
2	2	LOWERC	9.8573E-01	:	9.8193E-01	:	9.7846E-01	:	9.7546E-01	:	9.7254E-01	:	9.7019E-01	:
3	3	UPPERL	1.7423E+00	:	1.8656E+00	:	1.9817E+00	:	2.0808E+00	:	2.1755E+00	:	2.2567E+00	:
4	4	LOWERL	1.7697E+00	:	1.8848E+00	:	1.9997E+00	:	2.0975E+00	:	2.1899E+00	:	2.2698E+00	:
5	5	UPPERF	1.2590E+00	:	1.1331E+00	:	1.0198E+00	:	9.1782E-01	:	8.2604E-01	:	7.4344E-01	:
6	6	LOWERRF	1.1859E+00	:	1.0673E+00	:	9.6060E-01	:	8.6454E-01	:	7.7808E-01	:	7.0027E-01	:
INTERNAL DV. ID.	EXTERNAL DV. ID.	LABEL	12	:	13	:	14	:	15	:	16	:	17	:
1	1	UPPERC	9.6874E-01	:										
2	2	LOWERC	9.6842E-01	:										
3	3	UPPERL	2.3159E+00	:										
4	4	LOWERL	2.3280E+00	:										
5	5	UPPERF	6.6909E-01	:										
6	6	LOWERRF	6.3025E-01	:										

*** USER INFORMATION MESSAGE 6464 (DOM12E)

RUN TERMINATED DUE TO HARD CONVERGENCE TO AN OPTIMUM AT CYCLE NUMBER =

12.



To view the final shape and stress distribution, import the model and results into Patran, click on Results and select the options as given in the Results form in [Figure 6-31](#) and [Figure 6-32](#).

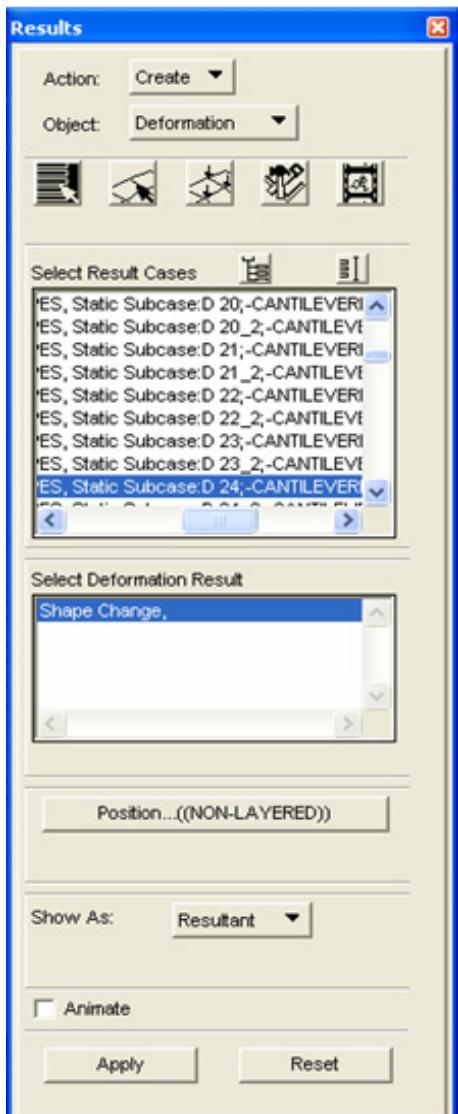


Figure 6-31 Viewing the final designed shape



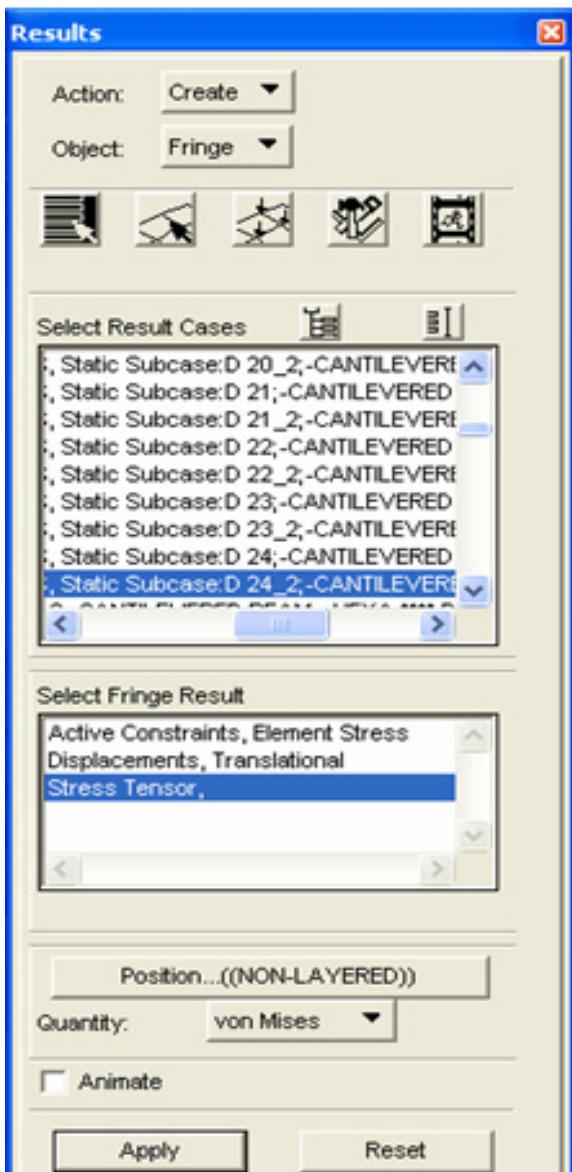


Figure 6-32 Visualizing the final stress distribution



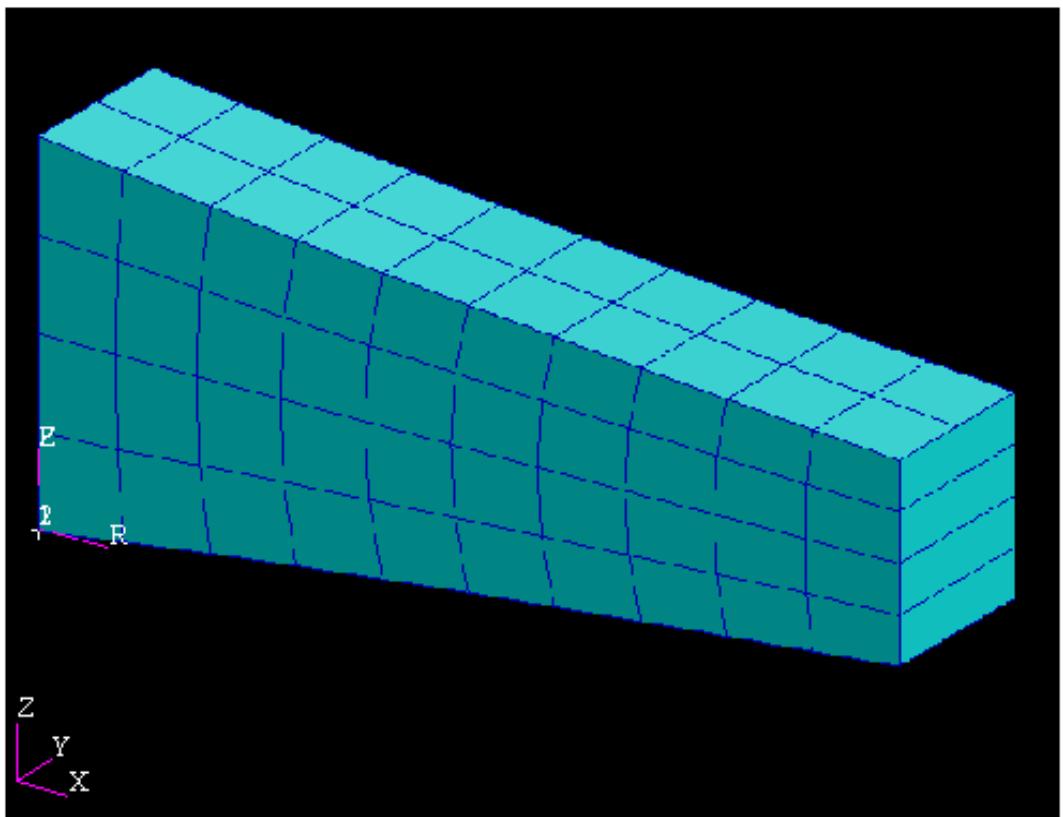


Figure 6-33 Solid Cantilever -- Final Shape



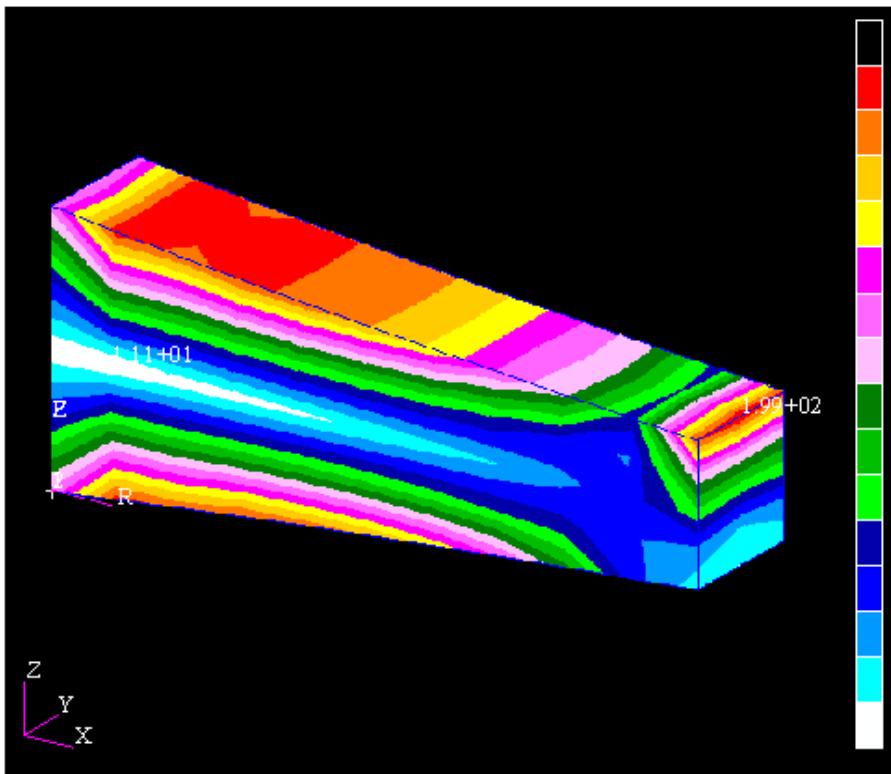


Figure 6-34 Solid Cantilever -- Final Stress Distribution

Special Punch in the Case of Shape Optimization

As described above, if shape optimization is being performed, a complete set of GRID Bulk Data entries will be punched every n-th design cycle where n is set using PARAM DESPCH. A special case occurs when the new mesh has become distorted to the extent that the resulting finite elements cannot be generated.

In this case, an invalid mesh message is produced by the EMG modals and, if it has not already been punched, the grid data for the last design whose mesh was valid as punched. This allows the user to start up again from this last valid mesh while perhaps changing some move parameters so that the mesh distortions do not occur.





7

Topology, Topometry and Topography Optimization

- Topology Optimization 372
- Brief Literature Review 373
- Bulk Data and Parameters 375
- Modeling Guidelines and Limitations 384
- Patran User Interface 387
- Application Examples 395
- Topometry Optimization 457
- Topography (Bead or Stamp) Optimization 468
- Toptomization 478



Topology Optimization

Introduction

In contrast to sizing and shape optimization for detail design, the layout and load-path study in the initial, conceptual design stage uses the Topology Optimization function. Topology Optimization can also be used to obtain rib patterns and weld distribution patterns. Topology optimization originally emphasized global design responses (such as structural compliance, eigenvalue, and displacements). Local stress limits on solid elements can be imposed to add practicality to the design proposal. In any case, it is recommended that topology optimization be used to generate a conceptual design proposal with emphasis on global design responses. Then a sizing and/or shape optimization can be performed based on the topology design proposal with emphasis on local design responses.

Features and Benefits

A number of features have been incorporated in SOL 200 for Topology Optimization as a continual effort to meet industry demands. The most common difficulties with Topology Optimization have been checkerboard effects, large number of small voids, introduction of large number of smaller members especially with mesh refinement, design proposals difficult to manufacture, and large computational cost with increasing variables. SOL 200 provides answers to each of these difficulties. The default filtering option helps overcome checkerboard effects. The minimum and maximum member size constraints offer the user control over the complexity of the design in terms of quantity of members. A large number of manufacturing constraints are available to ensure manufacturability, such as symmetry constraints, including cyclic symmetry, extrusion constraints for uniform thickness along draw direction, and single and two die casting constraints prevent cavities along die movement. Stress constraints on solid elements impose practical limits on the design.

Additional, features are available in topology optimization which increase the flexibility for the designer. Among these are setting multiple mass targets to different subdomains and combined size/shape and topology variables in a single job.



Brief Literature Review

The solution to the Topology Optimization problem is ill-posed in the sense that the design tends to a configuration with an unbounded number of microscopic holes rather than a small number of macroscopic holes. This suggests that the design will not generally converge to an optimum as the mesh is refined. There are two alternative ways for generating a well-posed Topology Optimization problem. In a procedure called relaxation (*Kohn and Strang* [Ref. 6.]) checkerboard designs are accommodated by extending the design space to include materials with periodic, perforated microstructures and then, using homogenization theory, to compute effective material properties. Alternatively, in a procedure called restriction, the design space is restricted to exclude checkerboard designs by imposing a perimeter constraint (*Ambrosio and Buttazzo* [Ref.7.]) or mesh independent filtering methods (*Sigmund* [Ref. 8.]).

In the homogenization-based approach, a composite material of a solid material and void is used instead of pure solid material. An example of a composite material is one composed of unit square cells with rectangular holes (*Bendsoe and Kikuchi* [Ref. 9.]), such that the variables defining the dimensions of the hole can vary between 0 to 1, thus covering the complete range of relative densities from zero (void) to one (solid). The microstructure of the composite can vary from one element in the mesh to the next. In this way, the Topology Optimization problem is converted from a 1-0 material distribution problem to a sizing problem. The theory of homogenization is used to replace the composite with a kind of equivalent homogeneous material and, in this way, relate the macroscopic material properties of the composite to its microstructure. The existence of a solution is guaranteed in this approach, but the solutions are hazy.

The Density Method approach, alternatively called the Power Law approach or Artificial Material approach (*Bendsoe* [Ref. 10.]), is based on the idea of convexification where an artificial material is used which is homogeneous. The density of the artificial material can vary between 0 and 1. The generalized material parameters are simply taken to be proportional to the relative density. A power law is used to relate the density with the material property.

$$\rho = \rho_0 \chi$$

$$E = E_0 \chi^p \quad (7-1)$$

where ρ_0 and E_0 are respectively the fully solid density and Young's modulus. A penalty factor p is introduced to enforce the design variable to be close to a 0-1 solution when $p>1.0$. The penalty factor p usually takes values between 2 and 5. This approach is simple and the optimum design consists of clear solid and void, but there is no suppression of local optima.

The homogenization approach has the advantage that the design can form rapidly along the lines of the force transmission path. The advantages of the density method are that it is more general and requires fewer design variables than the homogenization approach.

Whatever the approach, once the problem is formulated, an optimization method has to be employed to solve for the optimum design. The common methods are Mathematical Programming methods, Optimality Criteria (OC) methods or Evolutionary Structural Optimization method.



Mathematical Programming methods are general and efficient with provided gradients. Mathematical Programming methods are able to deal with many thousands of design variables and numerous constraints.

Optimality Criteria methods are indirect methods of optimization, unlike Mathematical Programming methods, which directly optimize the objective function. OC methods attempt to satisfy a set of criteria related to the behavior of the structure. Starting from a feasible point in the design space and using an iterative scheme based on an OC derived from *Kuhn-Tucker* conditions, the design variables are updated at each iteration so that the design gradually moves towards the optimum point. Though these methods are efficient for problems with a large number of design variables and few constraints, inclusion of more constraints slows down the process, and for general objective functions they do not work at all.

Evolutionary Structural Optimization (*Xie and Steven* [Ref. 11.]) is based on the intuitive concept that if understressed material is gradually removed from the design domain, an optimal, i.e. least weight, topology will be reached. This concept leads to a rejection criteria based on local stress level, where low stressed material is assumed to be underutilized and is removed. However, in a brief note (*Zhou and Rozvany* [Ref. 12.]) it has been shown that ESO's rejection criteria may result in a highly non-optimal design.

The Density method is used in MSC Nastran for Topology Optimization. As described earlier, the design variables are the normalized material density of each designed element. This approach has fewer design variables compared to the homogenization approach and is more general. The power law penalization on Young's Modulus E is used to achieve a 0-1 density distribution.

The original optimization problem is solved by solving a series of explicit approximate problems. Density and Young's Modulus are used as intermediate design variables.

For topology design sensitivity analysis, the adjoint method is available to compute efficiently the gradients of the topology objective and constraints with respect to topology design variables. The optimization algorithm used is MSCADS(method=4 SUMT) or IPOPT. All existing analysis types (statics, normal modes, buckling, frequency response, transient response, complex eigenvalue, static aeroelasticity, and flutter) are supported. Stress constraints can be applied to solid elements in static analysis.



Bulk Data and Parameters

TOPVAR

Topological Design Variables

To select a topologically designable region, the user needs to specify a group of elements. All elements referencing a given property ID are made topologically designable with the Bulk Data entry TOPVAR referencing that property ID. Topology design variables are automatically generated with one design variable per designable element. The manufacturability constraints are then applied on all elements referencing the given property ID.

Format:

TOPVAR	ID	LABEL	PTYPE	XINIT	XLB	DELXV	POWER	PID	
	“SYM”	CID	MS1	MS2	MS3	CS	NCS		
	“CAST”	CID	DD	DIE	ALIGN				
	“EXT”	CID	ED	ALIGN					
	“TDMIN”	TVMIN	TVMAX						
	“STRESS”	STLIM							

Example:

TOPVAR	1	TSHEL	PSHELL	.3			4	1	
	TDMIN	5.0	10.0						
TOPVAR	2	PSOLID	PSOLID					1	
	STRESS	5.0							

1	2	3	4	5	6	7	8	9	10
TOPVAR	2	PS1	PSOLID	0.3				10	
	SYM	5	XY	ZX					
	CAST	5	X	2					
	TDMIN	0.6							

Descriptor	Meaning
ID	Unique topology design region identification number. (Integer > 0)
LABEL	User-supplied name for printing purpose. (Character)
PTYPE	Property entry name. Used with PID to identify the elements to be designed. (Character: “PBAR”, “PSHELL”, “PSOLID”, etc.)



Descriptor	Meaning
XINIT	Initial value. (Blank or Real, $XLB < XINIT \leq 1.0$; Default = Blank). Typically, XINIT is defined to match the mass constraint on DRESP1=FRMASS, so the initial design does not have violated constraints. In this case, the default is set to the constraint value. If the mass (DRESP1=FRMASS or WEIGHT) is the objective, the default is 0.9. The default of XINIT is 0.6 for the other cases.
XLB	Lower bound to prevent the singularity of the stiffness matrix. (Real; Default = 0.001)
DELXV	Fractional change allowed for the design variable during approximate optimization. See Remark 3. (Real > 0.0 ; Default = 0.2)
POWER	A penalty factor used in the relation between topology design variables and element Young's modulus. (Real > 1.0 ; Default = 3.0). $2.0 \leq POWER \leq 5.0$ is recommended.
PID	Property entry identifier. This PID must be unique for PIDs referenced by other TOPVAR, DVPREL1 and DVPREL2 entries. Topology and sizing variables cannot share the same properties. (Integer > 0)
“SYM”	Indicates that this line defines symmetry constraints.
CID	Rectangular coordinate system ID used for specifying manufacturing constraints. See Remark 4. (Blank or Integer > 0 ; Default = blank)
MSi	Mirror symmetry plane. See Remark 5. & 7. (Character, ‘XY’, ‘YZ’, or ‘ZX’)
CS	Cyclic symmetry axis. (character X, Y, Z). See Remark 12.
NCS	Number of cyclic symmetric segments in 360 degrees (Integer > 0). See Remark 9.
“CAST”	Indicates that this line defines casting constraints (i.e., die draw direction constraints). See Remarks 6., 7., 8., and 10.
DD	Draw Direction. DDi=X, Y, Z or X-, Y-, Z- for a single die option (DIE=1) where X-, Y-, Z- indicates the opposite direction of X, Y, and Z respectively. DDi=X, Y, and Z for two die option (DIE =2) (Character)
DIE	Die Options. (Blank or integer 1 or 2; Default = 1) <ul style="list-style-type: none"> = 1 (or blank). A single die will be used and the die slides in the given draw direction (i.e., material grows from the bottom in the draw direction) = 2. Two dies will be used and the dies split apart along the draw direction (i.e., material grows from the splitting plane in opposite direction along the axis specified by the draw direction DDi. The splitting plane is determined by optimization)
ALIGN	Indicates whether the designed property finite element mesh is precisely aligned with the draw direction or extrusion direction. (Character: “YES” or “NO” or Blank; Default = blank = “NO”) See Remark 10.



Descriptor	Meaning
“EXT”	Indicates that this line defines extrusion constraints (i.e., enforce constant cross-section) See Remark 6. and 7.
ED	Extrusion direction. (Character, X, Y, or Z)
“TDMIN”	Indicates that this line defines a minimum and/or maximum member size., See remarks 11. and 12.
TVMIN	Minimum member size. See Remarks 11. and 12. (Real \geq 0.0 or blank)
TVMAX	Maximum member size. See Remarks 11. and 12. (Real $>$ TVMIN or blank)
“STRESS”	Indicates that this line defines a stress limit.
STLIM	von Mises stress upper bound. See Remark 13.. (Real >0.0)

Remarks:

1. The topologically designable element properties include PROD, PBAR, PBART, PBEND, PBEAM, PBEAML, PSHELL, PSHEAR, PSOLID, and PWELD. Multiple TOPVAR's are allowed in a single file. Combined topology, topography (BEADVAR), topometry (TOMVAR) sizing, and shape optimization is supported in a single file. However, TOPVAR cannot be used with DVMREL1 and DVMREL2 entries.
2. All designed element properties must refer to a MAT1 entry; therefore, a PCOMP/PCOMPG cannot be used as designed property in topology optimization. PCOMP/PCOMPG's can be used as non-designed properties in a topology optimization job.
3. If DELXV is blank, the default is taken from the specification of DELX parameter on the DOPTPRM entry.
4. Only CORD1R and CORD2R can be used as a referenced coordinate system to specify topology manufacturing constraints. Only one reference coordinate system CID is allowed for each TOPVAR entry.
5. One, two or three different mirror symmetry planes can present (such as MS1=XY, MS2=YZ, and MS3=ZX).
6. Casting (“CAST”) and Extrusion (“EXT”) manufacturability constraints can be applied to PTYPE=“PSOLID” only. Casting constraints cannot be combined with extrusion constraints for the same TOPVAR entry.
7. Some symmetry constraint types can be combined with casting or extrusion constraints. The referenced coordinate system CID must be the same for the combined constraints. Some possible combinations are:
 - For “EXT” constraints, possible combinations are (ED=X, MSi=XY, and/or ZX or CS=X), (ED=Y, MSi=YZ, and/or XY or CS=Y), (ED=Z, MSi=ZX, and/or YZ or CS=Z).
 - For “CAST” constraints, possible combinations are (DD=X or X-, MSi=XY and/or ZX or CS=X), (DD=Y or Y-, MSi=YZ and/or XY or CS=Y), (DD=Z or Z-, MSi=ZX and/or YZ or CS=Z).



8. For two dies option (DIE=2), the splitting plane is optimized. For a single die DIE=1, the parting plane is the bottom surface of the designed part in the draw direction.
9. The first symmetry segment starts at the X-axis when CS = Z (at Z-axis when CS = Y, and at the Y-axis when CS = X). One cycle symmetry can be combined with one mirror symmetry constraint as long as the axis of cyclic symmetry is normal to the plane of mirror symmetry. For example, MSi = YZ and CS = X, MSi = XZ and CS = Y, and MSi = XY and CS = Z. This feature can also be used for < 360 degrees but NCS must be given in 360 degrees.
10. It is recommended to use aligned mesh for casting property due to smaller tolerance used.
11. Without a TDMIN continuation line, the minimum member size constraint is taken from the specification of TDMIN parameter on the DOPTPRM entry. This option is applied on 2 and 3 D elements only. Minimum member size constraints can be used with "SYM", "CAST", and "EXT" constraints.
12. TVMIN and TVMAX are dimensional quantities. A guideline is that TVMIN be at least three times a representative element dimension. TVMAX must be greater than TVMIN and it is recommended that it be twice as big. If TVMAX is blank, no maximum member size is imposed. It is recommended that TVMIN always be used when TVMAX is specified.
13. "STRESS" limits can only be used for PTYPE=PSOLID. The stress constraints apply to all solid elements in both designed and non-designed regions. All TOPV AR entries must have the same STLIM.
14. The TOPVAR entry cannot be used with thermal loads.
15. For normal mode topology optimization, lower and higher mode may switch during optimization. This often occurs while maximizing or constraining the first eigenfrequency. This leads to a diverging solution. A workaround is using the mean value of a few of the lowest eigenfrequency (3~6) by DRESP2.

Bulk Data Entry DRESP1

Response Types FRMASS and COMP

While all DRESP1 response type are available for Topology Optimization, two response types (FRMASS and COMP) are available exclusively for Topology Optimization. The COMP response permits the specification of a compliance value as a design response. Compliance is simply the product of the displacement times the applied load and is typically used as an objective in a Topology Optimization design task to maximize structural stiffness in a static design problem. The FRMASS response indicates a target mass fraction that typically is used as a constraint in a Topology Optimization task; (e.g., minimize the compliance of the structure while limiting the mass to 40% of the mass of the original structure). It is not associated with a particular analysis type.

The COMP and FRMASS response types are provided to facilitate the specification of the classical Topology Optimization task of minimizing the compliance of a loaded structure while limiting the mass to some percentage of the maximum allowable amount. These responses can be applied generally so that the COMP response could lead to a constraint and the minimization of FRMASS could be an objective. The response attribute table for these two response types is given below:



Table 7-1 Responses for Topology Optimization

Response	Response Attributes		
Type (RTYPE)	ATTA	ATTB	ATT1
COMP (Remark 1)	BLANK	BLANK	
FRMASS (Remark 1,2)	BLANK	BLANK	BLANK or Property ID

Remarks:

1. RTYPE=COMP (compliance of structures = $P^T u$) and FRMASS (mass fraction of designed elements) entries are used for Topology Optimization or combined topology, sizing/shape optimization.
2. RTYPE=FRMASS is the mass divided by the mass calculated if all topology design variables are 1.0. FRMASS is calculated for topologically designed elements only. FRMASS = 1.0 if all topology design variables are 1.0. ATTi=Blank is for total mass fraction. ATTi=PID is the mass fraction for topological designed property PID.

Stress Constraints

Use of the DRESP1 to impose stress constraints in topology optimization is not a viable option in the normal case because the sensitivity calculation is prohibitive with the large number of design variables. Stress constraints can be imposed on the TOPVAR entry discussed above using the STLIM parameter. Note that this constraint is only used with PSOLID elements and it applies to designed and undesigned elements. If there are multiple TOPVAR entries, the STLIM value must be the same on each.

Design Optimization Parameters (DOPTPRM)**TCHECK and TDMIN**

Two design optimization parameters TCHECK and TDMIN are available on the DOPTPRM Bulk Data entry for overriding default values of parameters used exclusively in Topology Optimization. Parameter TCHECK is used to turn on a filtering parameter that is used to prevent a checkerboard pattern from being produced. The default value of 1 turns on the filtering and is recommended. TCHECK=0 turns off filtering. The other parameter, TDMIN, is used to achieve mesh independent solutions and control the size of members in the topology optimized design; that is, to prevent achieving a final design that is characterized by thin disjoint fibers that are impractical from a manufacturing standpoint. Design variable elements that are within a distance of TDMIN from an element with a design variable close to 1.0 are filtered to ensure they are not small. Note that the value of TDMIN is problem-size dependent but not mesh-size dependent. The default for TDMIN is 0.0 (no filtering). The descriptions of these parameters are given in [Table 7-2](#).



Table 7-2 DOPTPRM Design Optimization Parameters

Name	Description, type, and Default value
TCHECK	Topology Filtering options (integer 1) -1 Automatic selection of filter or density constraint (Default) 1 Filtering algorithm is on for Topology Optimization 2 Density Constraint method is On 0 No filtering algorithm
TDMIN	Topology minimum member diameter (real ≥ 0.0) in the basic coordinate system. Default =0.0 (i.e., no minimum member size control). It is recommended at least three times a representative element dimension. This option is applied on 2 and 3 D elements only.

Note that TDMIN values on the TOPVAR entry overwrite the DOPTPRM value.

In addition, several existing DOPTPRM parameters have different default values for Topology Optimization as opposed to Sizing/Shape optimization, as shown in [Table 7-3](#). For combined sizing/shape and Topology Optimization problems, the default values for Topology Optimization are imposed.



Table 7-3 Default Values for DOPTPRM Design Optimization Parameters

Parameter	Sizing/Shape	Topology
DESMAX	5	30 or 60 with minimum member size
DELX	0.5	0.2
DXMIN	0.05	1.0E-5
CONV1	.001	.005 (when a clear topology solution is obtained)

As a final comment on the DOPTPRM parameters described in [Chapter 4: Optimization Parameters](#), the definition of the P2 parameter that controls the amount of print that occurs at design cycles specified by P1 has a different meaning in a Topology Optimization context. For sizing and shape optimization, design variables are printed for value of P2 = 1 (or if 1 is included in the sum of the options, that is, 3, 5, 7, 9, 11, 13 and 15). Since a Topology Optimization task can easily result in thousands of design variables, this would not be a viable option for most problems. Instead, design variable prints are turned off unless a P2 value greater than 8 (that is 9, 11, 13 or 15) is specified. The "Comparison between Input Property Variables from Analysis and Design Models" print never occurs in Topology Optimization.

Output for Topology Optimization

Output for the two responses, compliance and fractional mass, and topology design variables are shown in [Figure 7-1](#). Also in this figure, the design variable history shows the external element ID associated with the internal design variable ID



----- COMPLIANCE RESPONSES -----						
INTERNAL ID	DRESP1 ID	RESPONSE LABEL	LOWER BOUND	VALUE	UPPER BOUND	
1	1	COMPL	N/A	1.4162E+02	N/A	
----- FRACTIONAL MASS RESPONSES -----						
INTERNAL ID	DRESP1 ID	RESPONSE LABEL	LOWER BOUND	VALUE	UPPER BOUND	
2	2	FRMASS	N/A	3.0000E-01	3.0000E-01	

***** SUMMARY OF DESIGN CYCLE HISTORY *****						
***** DESIGN VARIABLE HISTORY *****						
INTERNAL DV. ID.	EXTERNAL ELEMENT ID	LABEL	INITIAL	:	1	:
1	1	TOPVAR	3.0000E-01	:	2.4000E-01	:
2	2	TOPVAR	3.0000E-01	:	2.4000E-01	:
3	3	TOPVAR	3.0000E-01	:	2.4000E-01	:
4	4	TOPVAR	3.0000E-01	:	2.4000E-01	:
5	5	TOPVAR	3.0000E-01	:	3.6000E-01	:
6	6	TOPVAR	3.0000E-01	:	3.6000E-01	:
7	7	TOPVAR	3.0000E-01	:	2.4000E-01	:
8	8	TOPVAR	3.0000E-01	:	2.4000E-01	:
9	9	TOPVAR	3.0000E-01	:	2.4000E-01	:
10	10	TOPVAR	3.0000E-01	:	2.4000E-01	:

Figure 7-1 Selected .f06 output for Topology Optimization

Special Punch Considerations for Topology Optimization

MSC Nastran topology optimization generates an element density distribution file jobname.des for all design cycles as a Patran element result format that is supported in the Patran postprocessor as well as other postprocessors.

For sizing and shape optimization, the DESPCH parameter specifies when the optimized Bulk Data entries are written to the PUNCH file. In Topology Optimization, DESPCH is used to specify when the topology optimized element density values are written to the topology element density history file jobname.des. This file can be directly read into Patran to display and animate the Topology Optimization results. [Figure 7-1](#) shows the format of an element density history file.



DESPCH < 0 Never

DESPCH = 0 at the last design cycle only (default)

DESPCH > 0 at every design cycle that is a multiple of DESPCH and the last design cycle. For example, if n=2 and the maximum number of design cycles is 5 (DESMAX=5 on the DOPTPRM entry), then, topology element density at design cycle 2, 4, and 5 are written in the punch file.

DESPCH1 >= 0 (default=0), write all (topology designed and non-designed) element density values to the topology element density history file jobname.des.

DESPCH1 < 0, write topological designed element density values to the topology element density history file jobname.des.

/DENSI/	←	Flag for element density file
1	←	Design cycle ID
10		
1	0.240	External element ID and density value
2	0.240	
3	0.240	
4	0.240	
5	0.360	
6	0.360	
7	0.240	
8	0.240	
9	0.240	
10	0.240	

Figure 7-2 Element Density History File jobname.des



Modeling Guidelines and Limitations

The quality of the results of a Topology Optimization task is a strong function of how the problem is posed in MSC Nastran. This section contains a number of tips that have been developed based on extensive testing of the capabilities.

Modeling Tips

1. A DRESP1=COMP is introduced to define the compliance of structures for Topology Optimization. The response is usually used as an objective to maximize structural stiffness in static analysis problems.
2. A DRESP1=FRMASS is introduced to define the mass fraction of topology designed elements. For Topology Optimization tasks, the DRESP1=FRMASS response is recommended to define a mass reduction target in a design constraint.
3. While FRMASS is calculated for topological designed properties only, RTYPE=WEIGHT computes total weight including all designed and non-designed parts. For combined topology and sizing/shape optimization problems, it is recommended that RTYPE=FRMASS be used for topological designed property mass reduction constraints and RTYPE=WEIGHT be used for the total mass reduction constraints.
4. The POWER field on the TOPVAR entry has a large influence on the solution of Topology Optimization problems. A lower POWER (less than 2.0) often produces a solution that contains large "grey" areas (area with intermediate densities 0.3 - 0.7). A higher value (greater than 5.0) produces more distinct black and white (solid and void) designs. However, near singularities often occur when a high POWER is selected.
5. A parameter TCHECK on DOPTPRM is used to turn on/off the checkerboard free algorithm. This default normally results in a better design for general finite element mesh. However, if higher order elements and/or a coarser mesh are used, turning off the filtering algorithm may produce a better result. For minimum member size control, the density constraint method is very efficient, especially when the mesh is fine and the predefined minimum member size is relatively large. This method also helps to get a checkerboard free solution. It is activated by TCHECK=2.
6. The parameter TDMIN on DOPTPRM is mainly used to control the degree of simplicity in terms of manufacturing considerations. It is common to see some members with smaller size than TDMIN at the final design since the small members have contributions to the objective. Minimum member size is more like quality control than quantity control. It is in general recommended that TDMIN should not be less than the length of 3 elements. Note that the TOPVAR entry has a "TDMIN" line that sets TV, which provides a minimum member size for each TOPVAR.
7. It is recommended to use default value of XINIT.
8. Maximum design cycle DESMAX=30 (as default) is often required to produce a reasonable result. More design cycles may be required to achieve a clear 0/1 material distribution, particularly when manufacturability constraints are used.



9. There are many solutions to a Topology Optimization problem, one global and many local optima. It is not unusual to see different solutions to the same problem with the same discretization by using different optimization solvers or the same optimization solver with different starting values of design variables.
10. To obtain a rib pattern by Topology Optimization, a core non-designable shell element thickness must be defined together with two designable element thicknesses above and below the core thickness. That is, add two designable elements for each regular element.
11. If some elements are disconnected on the final topology design proposal this may be because, the mass target may be too small to fill the design space.
12. It is recommended that a baseline Topology Optimization job (without any manufacturability constraints) be carried out before a Topology Optimization solution with manufacturability constraints is done. Benefits are: (a) a Topology Optimization without restriction may result in a better design; (b) The design proposal from the no restriction run may give some hints for imposing manufacturability constraints.
13. Topology Optimization with manufacturability constraints often needs more material to fill the design space. Therefore, the design with manufacturability constraints usually requires a relatively bigger mass target (less material savings) than the one without manufacturability constraints.
14. It is recommended to use aligned mesh for casting property due to smaller tolerance used.
15. If multiple mass targets (multiple DRESP1=FRMASS) are used, it is recommended that each TOPVAR's initial value XINIT matches its corresponding mass target.
16. The cyclical symmetry constraints can also be used for rotational parts <360 degree. In addition, the starting surface must be XY plane for cyclical symmetry axis CS=X, YZ plane for CS=Y; ZX plane for CS=Z respectively. The number of cyclical symmetric segments (NCS) must also be defined in 360 degrees for this case. For example, if a 90 degree rotational part has 3 segments, NCS must be set to NCS=12 in 360 degree.
17. Patran can smooth, remesh, and generate IGES files for 2D topology designs and smooth 3D topology designs.
18. Patran can support Topology Optimization with manufacturability constraints.
19. It is recommended that the IPOPT optimizer option be utilized for Topology Optimization and for sizing and shape optimization problems with thousands of design variables. This is the default.
20. Numerical problems often occur when solving a Topology Optimization task. The nature of the problem depends on element type, number of elements, optimization algorithm and so on. One frequent numerical problem is the so-called checkerboard effect. Checkerboard-like material distribution pattern is observed in the Topology Optimization of continuum, especially when first order finite elements, such as CQUAD4, are employed to analyze structural responses. It has been shown that the Checkerboard-like phenomenon is caused by the finite element formulation. The problem occurs because the checkerboard has an artificially high stiffness compared with a structure with uniform material distribution. The easiest way to decrease the checkerboarding effect is to use higher order elements (such as CQUAD8). This however increases the CPU-time considerably. Another closely related phenomenon is mesh-dependent solutions. It is seen that a more detailed structure is found by increasing the number of elements. The rationale of making a



finer finite element mesh is to get a better finite element solution. However, this finer meshing tends to have an increasing number of members with decreasing size. This more detailed topology solution creates a problem from a manufacturing point of view. In SOL 200, filtering algorithms are used to promote a checkerboard-free and mesh independent topology optimized solution and TDMIN is used to limit the increasing number of members.

21. The CASI solver is strongly recommended for solid elements topology problems for efficiency.
22. In many applications, loads or boundary conditions are related to assembly. It cannot be changed. In this case, it is the best for users to model those elements associated with loads or boundary conditions with different property IDs. Thus, those PIDS can be excluded from design space (TOPVAR entry NOT references those PIDS).

Limitations

- Elements referencing the composite property PCOMP entry cannot be designed.
- Superelements are not supported.
- Thermal loads are not supported.
- CASI solver is limited to compliance minimization Topology Optimization problems only.
- Although combined topology and sizing optimization is supported, TOPVAR and DVPREL1/2 entries cannot reference the same property ID (PID).

RTYPE=COMP can be used even with pure sizing problems provided that Adjoint Method is used for sensitivity calculation. However, RTYPE=COMP cannot be used in direct sensitivity calculation.

RTYPE=TOTSE (total strain energy) can be used in direct sensitivity calculation. RTYPE=FRMASS is not supported in pure sizing problems even for the Adjoint Method.

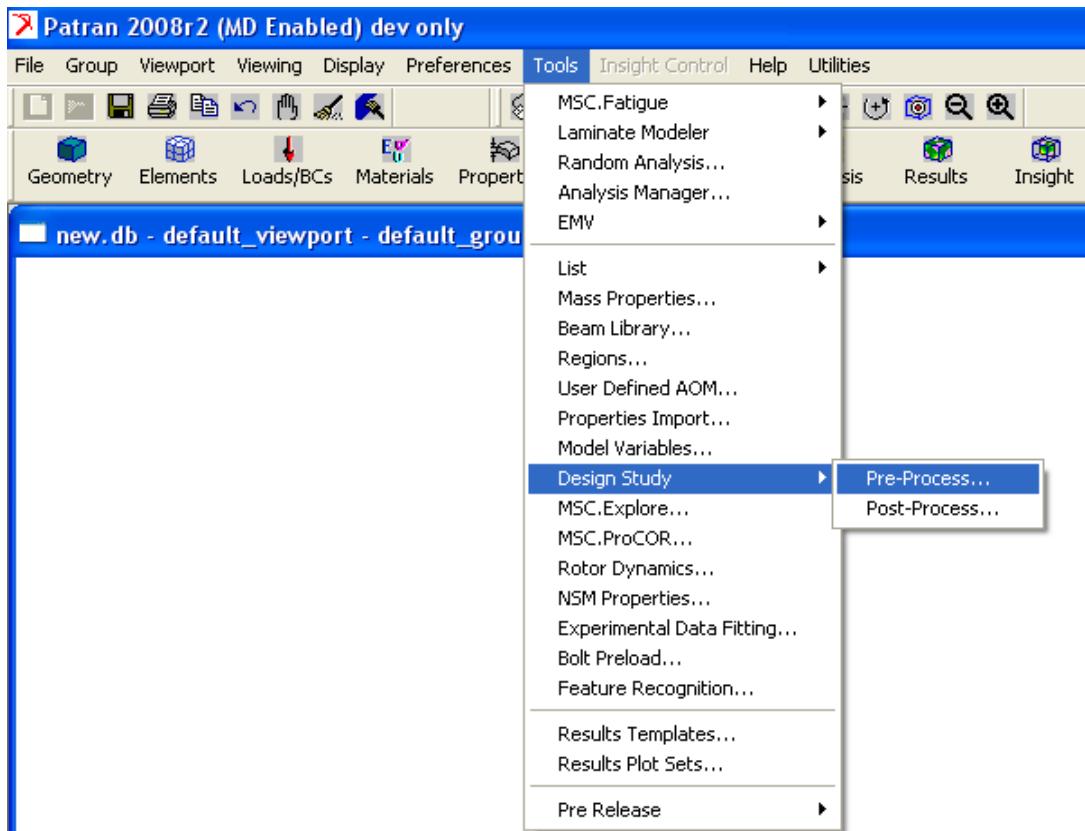


Patran User Interface

All input data used in Topology Optimization analysis can be generated in Patran. This support includes all the features like combined size and Topology Optimization, multiple mass targets, manufacturing constraints, multidisciplinary optimization and output controls.

General Topology Optimization Preprocessing

From the Main Menu select Tools -> Design Study -> Preprocess



In the Pre-Process form, select **Action**: Create; **Object**: Design Variable; **Type**: Topology to create TOPVAR.



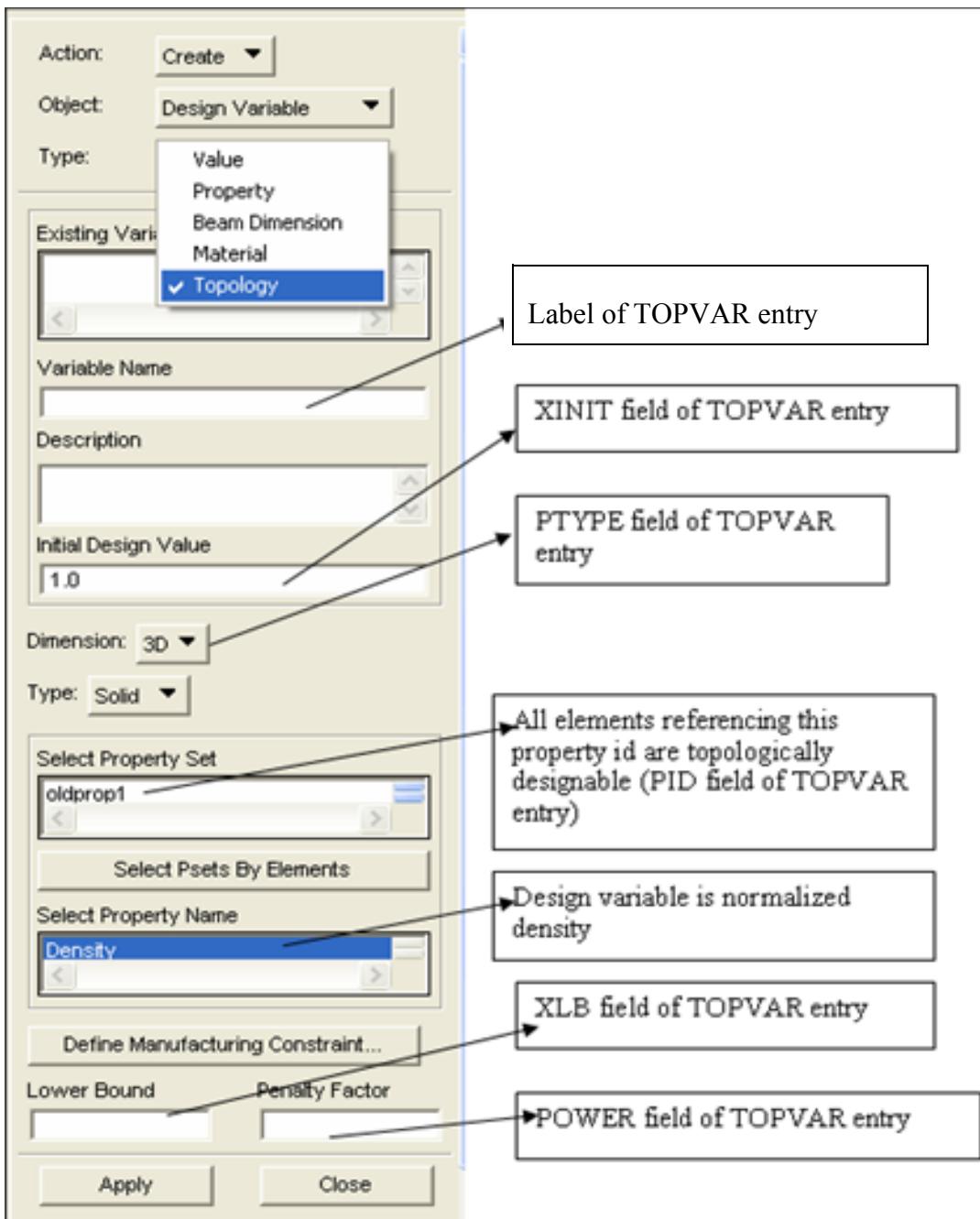


Figure 7-3 Patran Pre-process



The Manufacturing Constraints can be added by clicking the **Define Manufacturing Constraints** tab and filling up the form which appears (see [Figure 7-4](#)).

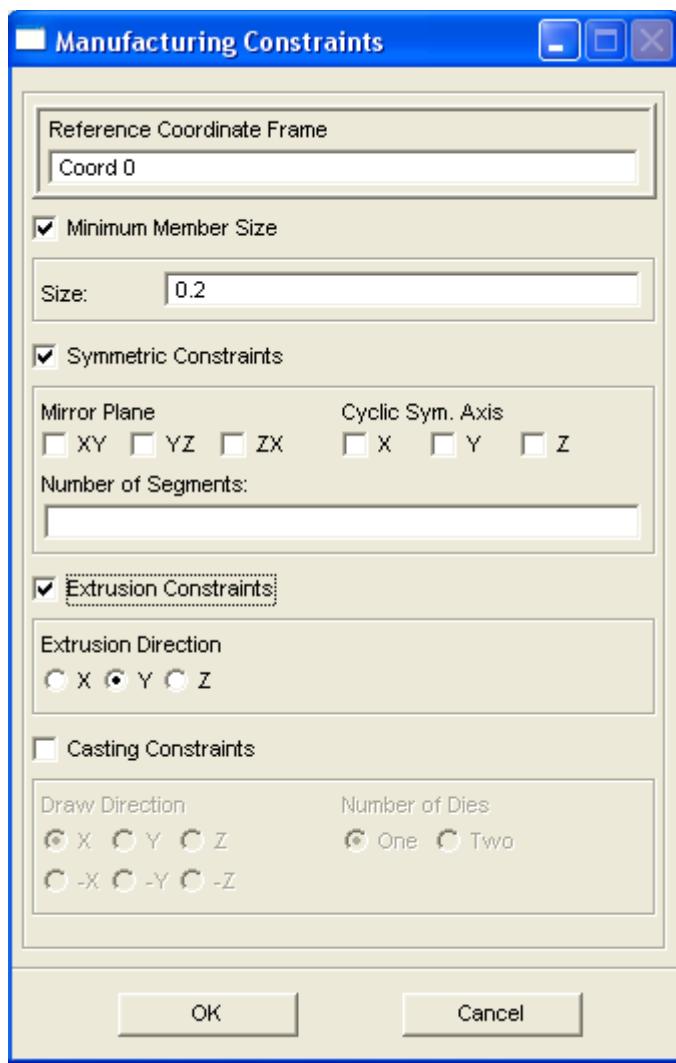


Figure 7-4 Creating Topology Variable

In the Pre-Process form, select **Action**: Create; **Object**: Objective; **Solution**: Linear Static; **Response**: Compliance; **Min/Max**: Minimize to create Compliance Minimization as the Objective. Then select **Action**: Create; **Object**: Constraint; **Solution**: Global; **Response**: Fractional Mass to create Constraint on Fractional Mass. The converse can be done for Fractional Mass minimization subject to Compliance constraint.



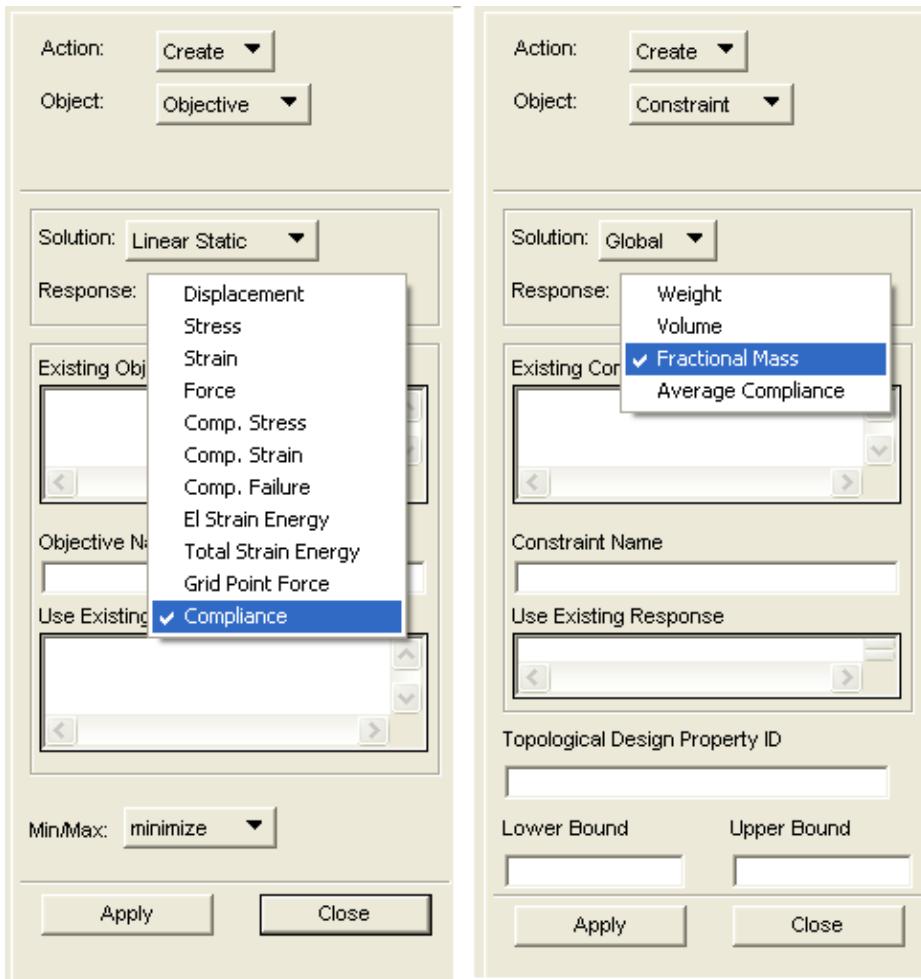


Figure 7-5 Creating Objective and Constraint

Note that when FRMASS (Fractional Mass) constraint is being created, there is provision to select a topologically designable region identified by a Property ID. Many such constraints can be created to allow for separate mass targets on different regions. This is the ATT1 field for RTYPE=FRMASS in the DRESP1 entry. Furthermore, the Lower Bound and Upper Bound boxes correspond to the LALLOW and UALLOW fields of the DCONSTR entry.

For other responses, analysis types and combined size and Topology Optimization, the procedure is just the same as for size optimization.

For Output Control, click **Analysis** and in the Analysis form select **Action: Optimize**; **Object: Entire Model**; **Method: Analysis Deck** and click on **Optimization Parameters**. This opens the Optimization Parameters form, which allows the user to select values for DOPTPRM parameters like P1, P2, and



DESMAX, and few parameters like CONV1, CONV2, DELP and DELX on clicking the **Advanced Optimization Parameters** tab.

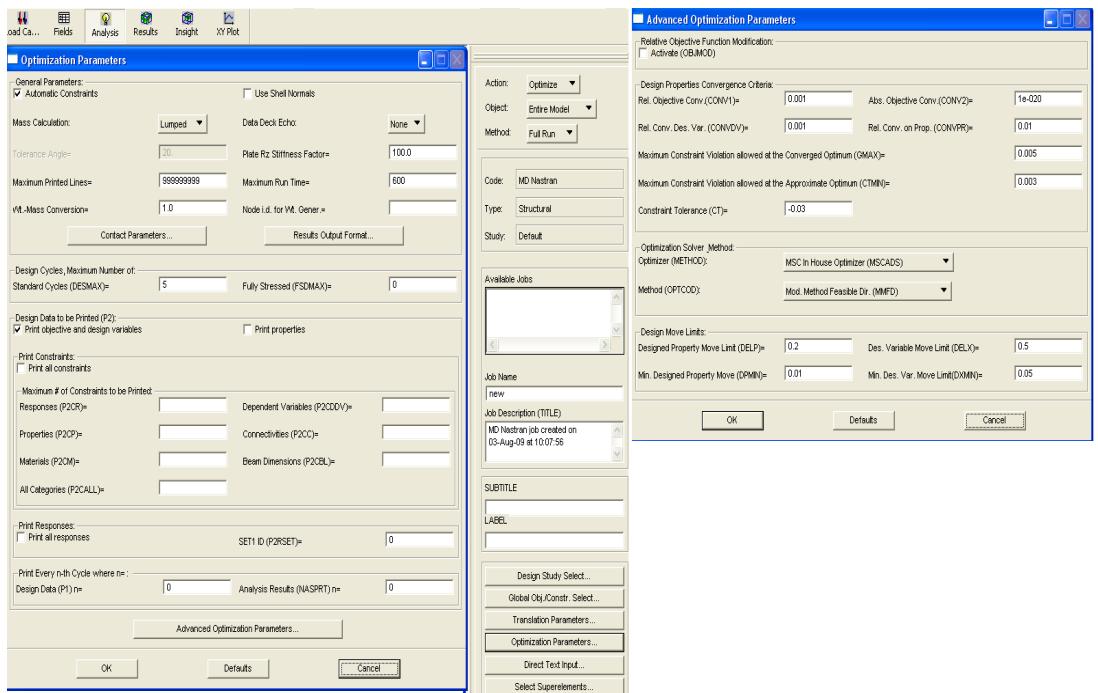


Figure 7-6 Setting Optimization Parameters

Patran Postprocessing

MSC Nastran produces a file with .des extension which contains the resulting optimal element density distribution. This file can be directly read in Patran to display the Topology Optimization results. Patran supports read and display of topology results, the smoothing/ remeshing of a topology design proposal for a new reanalysis and the generation of IGES files for a topology design proposal for CAD systems.

From the Main Menu select **Tools → Design Study → Postprocess**



Then in the Post-Process form select **Action**: Read Results, then click the **Select Results File** tab to Browse through the list of available .des files, select the required file and then click **Apply**.

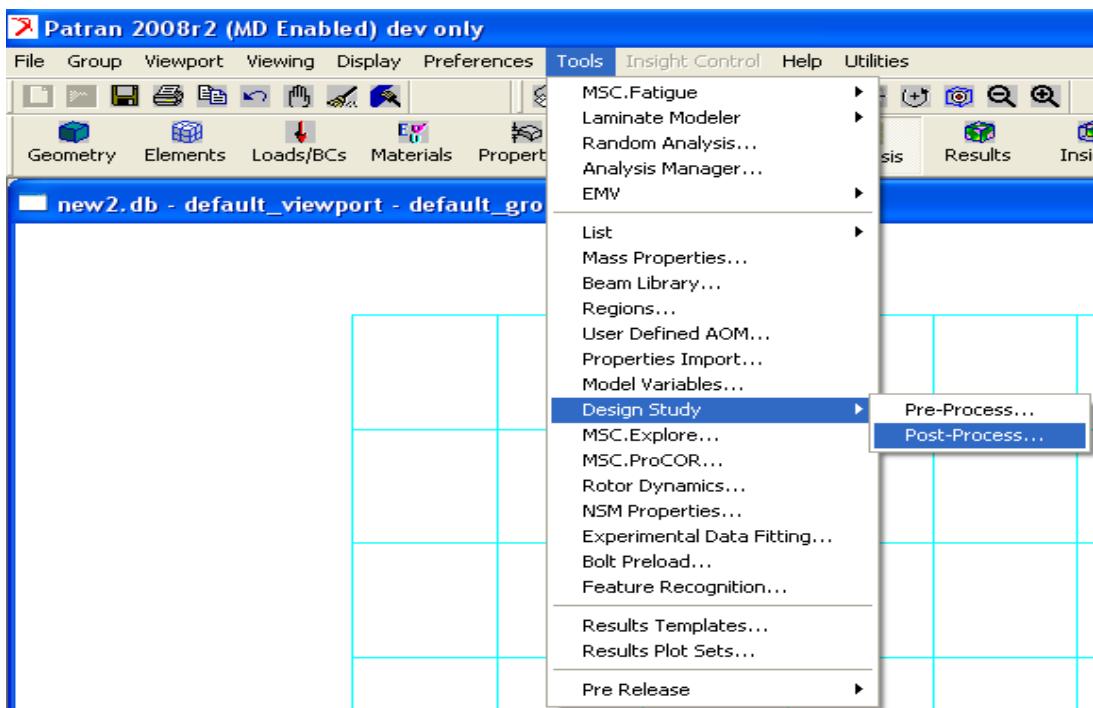


Figure 7-7 Patran Post-process

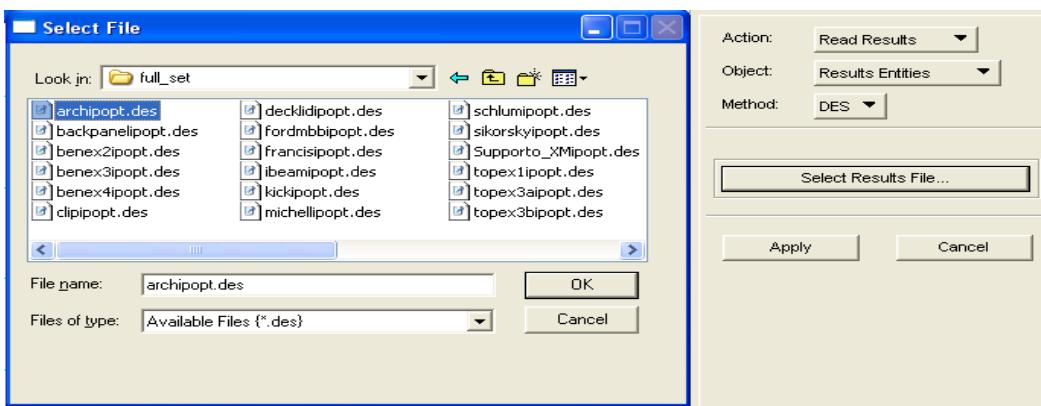


Figure 7-8 Reading Density Distribution

To see the results, select **Action**: Display Results and the required design cycle from **Select Result Case** list box, input a Threshold limit (density value below which the density will be treated as 0 or void), select the Fringe check box (if density contours are desired). Click **Apply**.



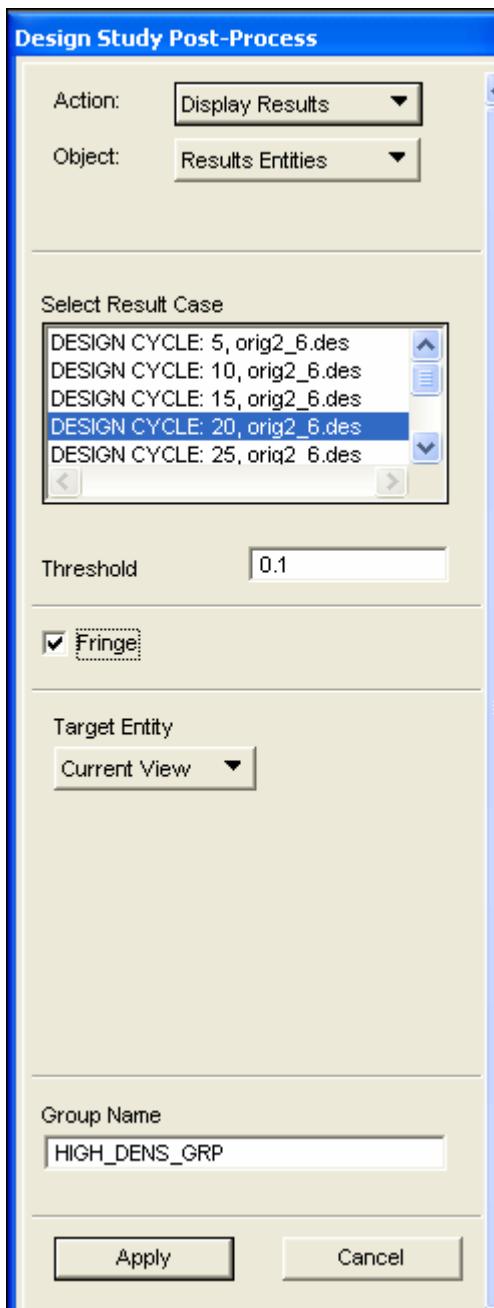


Figure 7-9 Displaying Density Distribution



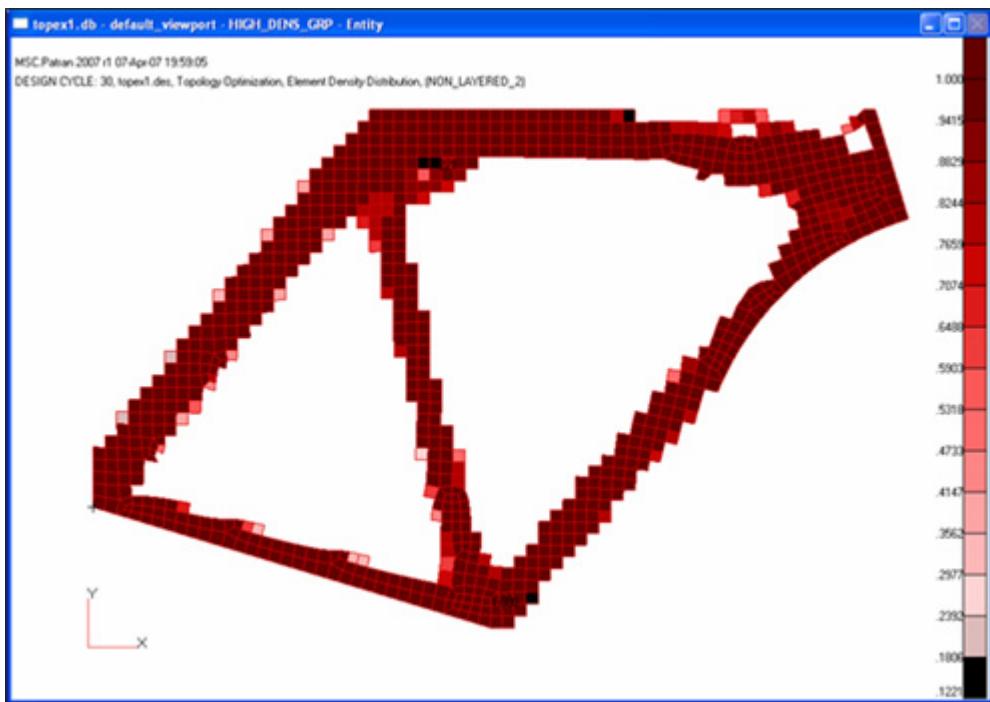


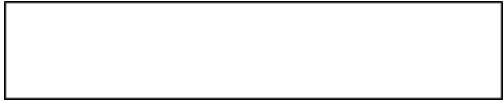
Figure 7-10 Density Distribution Plot



Application Examples

Bridge Example

Summary

ATTRIBUTE	VALUE
Title	Bridge
Topology Optimization features	Compliance minimization Mass target Mirror symmetry constraints
Geometry	 Length = 40 and width = 20 Thickness = 0.1
Material	E = 2.0E+5 Pa, $\mu=0.3$
Analysis	Static analysis
Boundary conditions	Supported on rollers at one point and fixed support at another point
Applied loads	A concentrated force = 10.0 N
Element types	4 node linear QUAD elements
Topology result	Material distribution

Introduction

A simply supported bridge example (model shown in Figure 5.1) is used to demonstrate (a) basic MSC Nastran Topology Optimization capabilities without manufacturing constraints (topoug1.dat in TPL) and (b) mirror symmetry constraints (topoug1a.dat in TPL). The structural compliance (i.e., total strain energy) is minimized with a mass target 0.4 (i.e., 60% material savings). The loading and boundary conditions are shown in [Figure 7-11](#). The structure is modeled with 3200 CQUAD4 elements. An additional objective of this exercise is to give a step-by-step procedure in Patran to (a) import the model data (b) setup a Topology Optimization problem (c) read the topology results (.des) file (d) display the density distribution (e) smooth the boundaries (f) create a surface over the smoothed topology proposal and (g) export the IGES file of the topology proposal.



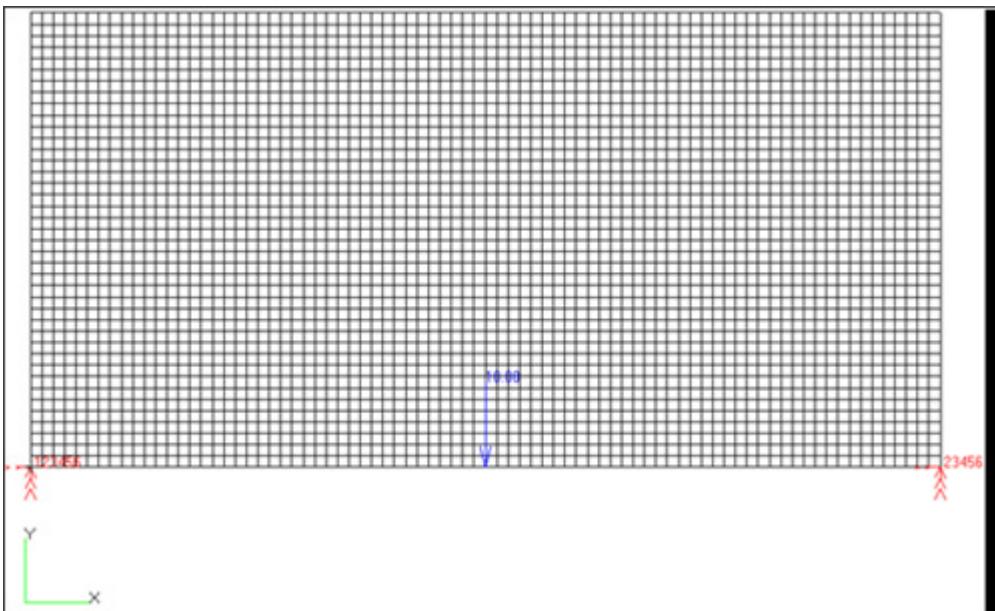


Figure 7-11 Bridge

Solution Requirements

This bridge example is widely used by academic and industrial researchers for Topology Optimization validation.

Design Model Description

Objective: Minimize compliance

Topology design region: PSHELL

Constraints: Mass target = 0.4 (i.e., mass savings 60%)

With or without mirror symmetry constraints

These solutions demonstrate:

- A distinct design can be obtained by MSC Nastran Topology Optimization with checkerboard free algorithm (as default)
- By using symmetry constraints in Topology Optimization, a symmetric design can be obtained regardless of the boundary conditions or loads.
- The smoothed topology proposal can be exported as an IGES file which can be used by any CAD system.



Optimization Solution

Basic Compliance Minimization

The input data for this example related to Topology Optimization model is given in [Listing 7-1](#). A Bulk Data entry TOPVAR =1 is used to define a topological design region. XINIT=0.4 on the TOPVAR entry matches the mass target constraint so that the initial design is feasible. The rest of the values on the TOPVAR entry are default values that are recommended for general Topology Optimization applications. Type one design responses DRESP1 = 1 and 2 identify compliance and fractional mass, respectively. DCONSTR= 1 specifies the mass target. DESOBJ=1 in Case Control Command selects DRESP1=1 entry to be used as a design objective (minimization as default) and DESGLB selects the design constraint DCONSTR= 1 to be applied in this Topology Optimization task.

Listing 7-1 Input File for Bridge Example

```

DESOBJ = 1
DESGLB = 1
SUBCASE 1
$ Subcase name : Default
SUBTITLE=Default
SPC = 2
LOAD = 2
ANALYSIS = STATICS
BEGIN BULK
DCONSTR 1      2           .4
TOPVAR,    1   ,     Tshell,    Pshell, .4, , , , 1
DRESP1    1       COMPL     COMP
DRESP1    2       FMASS     FMASS

```

[Figure 7-12](#) shows the topology optimized result. This optimal design is very clear without any checkerboard effect.



Figure 7-12 Bridge Topology Design

Mirror Symmetric Constraints

Since the loads applied on the bridge are not symmetric, the topology optimized design [Figure 7-12](#) is not symmetric about z-x plane. The bridge is employed again to demonstrate the mirror symmetric constraint capability that enforces the design to be symmetric about a given plane. This entire exercise is taken up using Patran.



Step 1: Importing the Bridge Model into Patran

- Create a new database in Patran. Click **File → Import**
- In the Import form select **Object: Model; Source: MSC Nastran Input**.
- Select the required data file from the browser. (This file contains the finite element model suitable for a static analysis lacking any topology related input.)
- Click **Apply**.

The Topology Optimization problem will be setup on this base model.

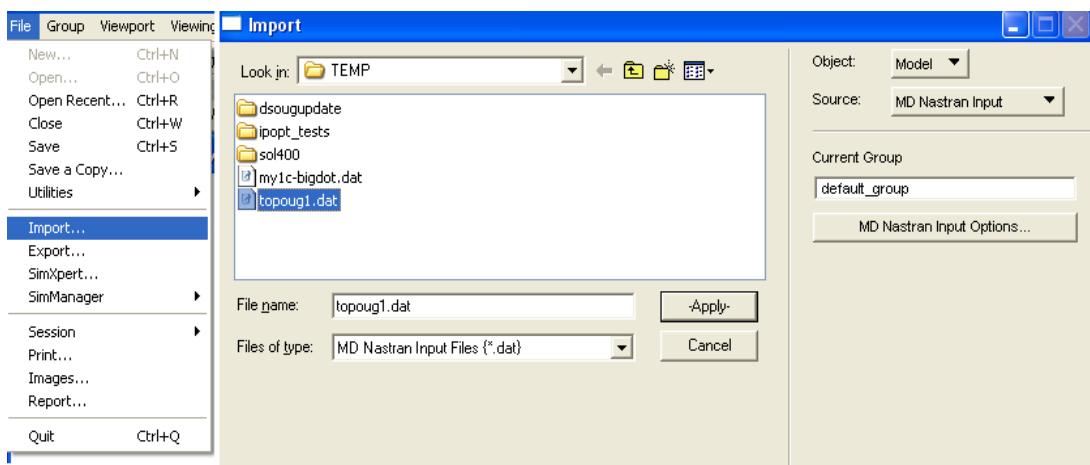


Figure 7-13 Importing Model into Patran

Step 2: Creating a Coordinate Frame for Defining Mirror Plane

A new Coordinate frame is created with its origin at the midpoint of the rectangular domain.

- Click **Geometry**.
- In the Geometry form, select **Action: Create; Object: Coord; Method: 3 point; Type: Rectangular**
- Enter [20 10 0] for Origin, [20 10 1] for a Point on Axis 3 and [21 10 0] for a Point on Plane 1-3
- Click **Apply**.

This rectangular coordinate frame will define the mirror plane for the symmetry constraint (see [Defining Design Domain and Manufacturing Constraints](#)).



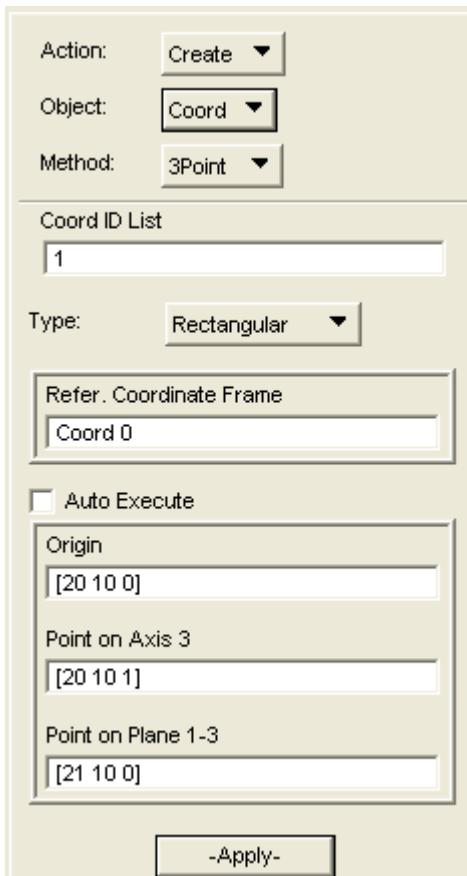


Figure 7-14 Creating Coordinate Frame

Step 3: Creating a Load Case

The imported model has the FE entities and force and support entries, so a Load Case has to be created combining the forces and supports.

- a. Click **Load Case**.
- b. In the Load Case form, select Action: Create; Type: Static
- c. Enter a name for the load case in the Load Case Name box.
- d. Click the **Input Data** tab.
- e. In the Input Data form, click all the entries in the Select Individual Loads/BCs list box.
- f. Click **OK**.
- g. Click **Apply**.



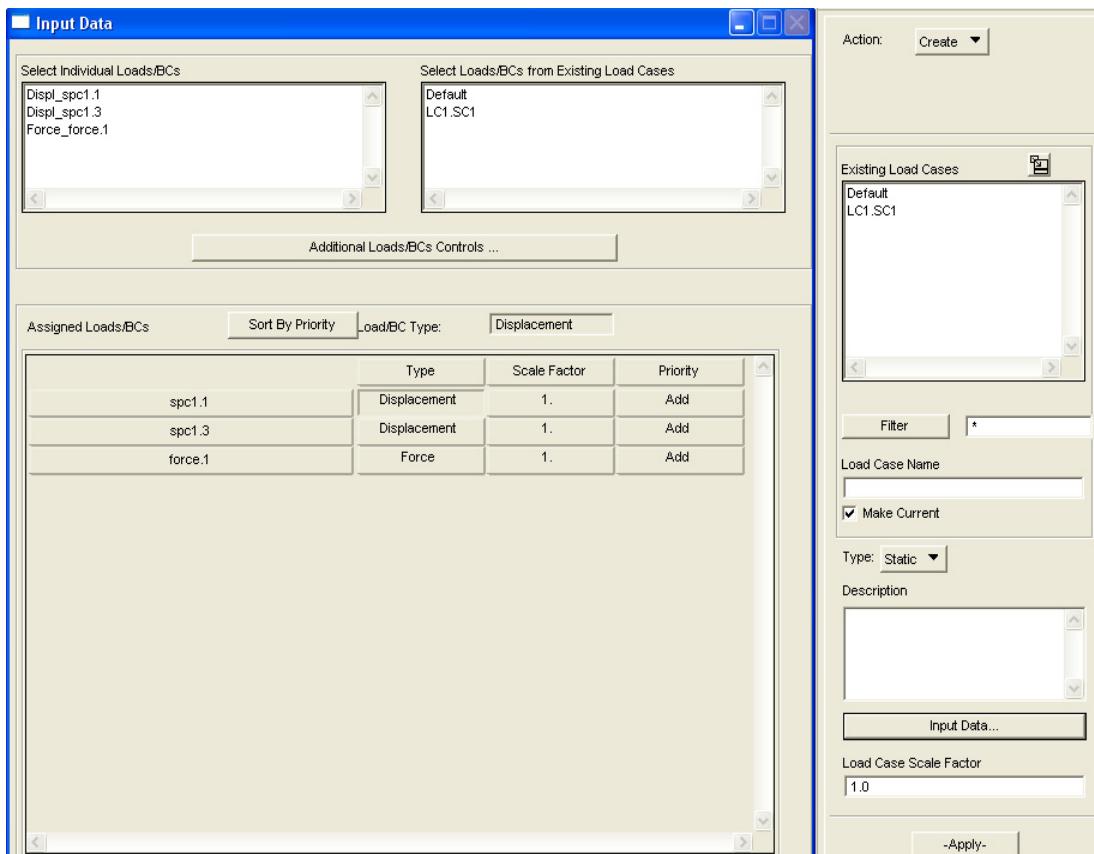


Figure 7-15 Creating Load Case

Step 4: Initiating Topology Toptimization:

- Click **Analysis**.
- In the Analysis form, select **Action**: Toptimize, **Object**: Entire Model, **Method**: Analysis Deck (Selecting **Method**: Full Run allows MSC Nastran to be called through the GUI itself.)



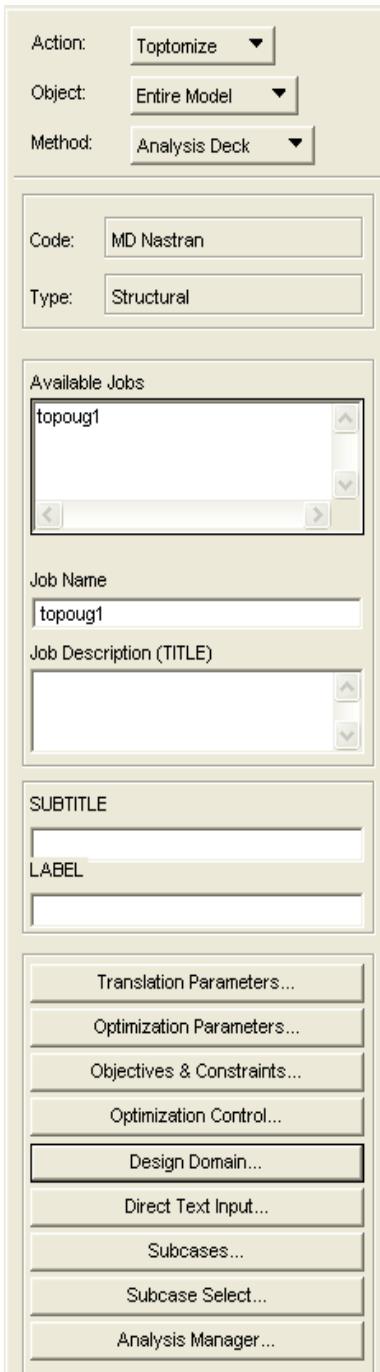


Figure 7-16 Initiating Topoptimize for Topology Optimization



Step 5: Defining Objectives and Constraints

- In the Analysis form, click the **Objectives and Constraints** tab. In the Objectives and Constraints form, select **Type**: Topology, **Objective Functions**: Minimize Compliance, **Constraint Target**: Mass Fraction. Enter value of Mass Target Constraint as 0.4. Click **OK**.

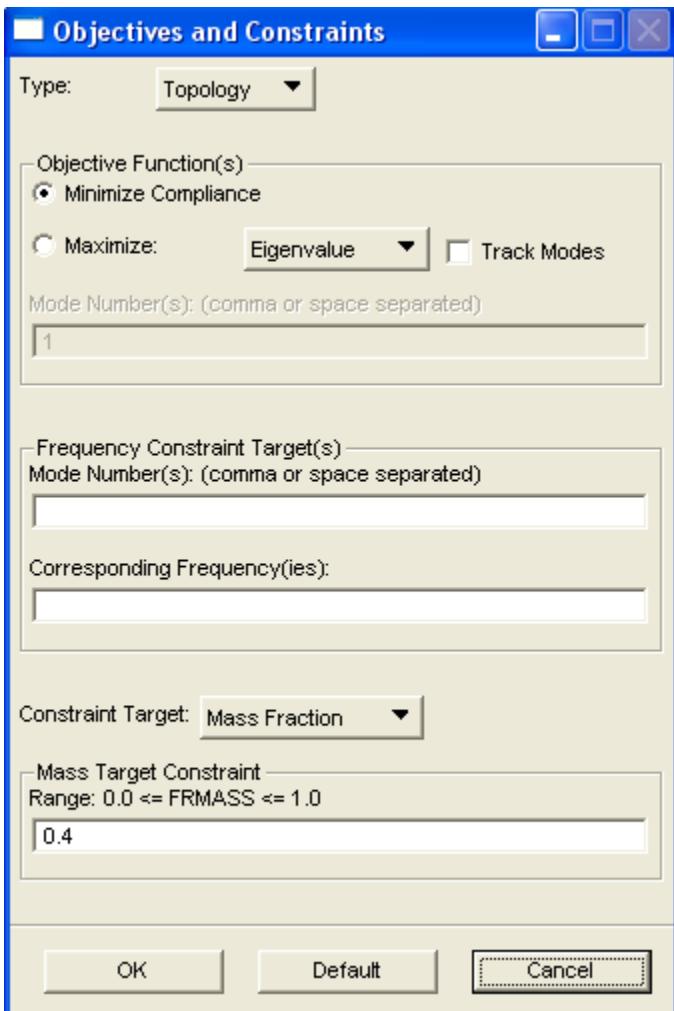


Figure 7-17 Creating Objectives and Constraints

Step 6: Defining Design Domain and Manufacturing Constraints

- Click the **Design Domain** tab. In the Design Domain form, click property name prop1 in Valid Properties then click the Manufacturing Constraints tab.
- In the Manufacturing Constraints form, enter Reference Coordinate Frame as Coord 1 (created in [Creating a Coordinate Frame for Defining Mirror Plane](#)). Select the the Symmetric Constraints and Mirror Plane ZX check boxes. Click **OK** to close the form.



- c. Click **OK** to close Design Domain Form.

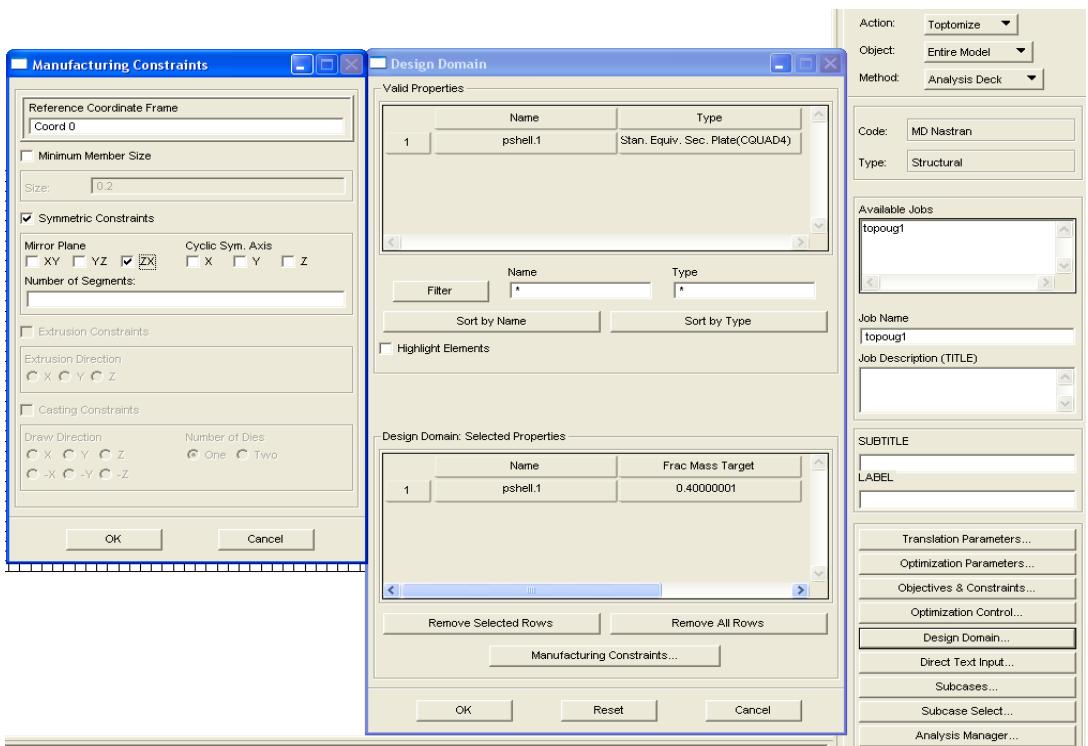


Figure 7-18 TOPVAR entry

This process results in generation of the TOPVAR entry as shown under:

Step 7: Defining Optimization Parameters

- Click the **Optimization Control** tab.
- Input 0.4 in Initial Design box in the Optimization Control Parameters form.
- Click **OK**.



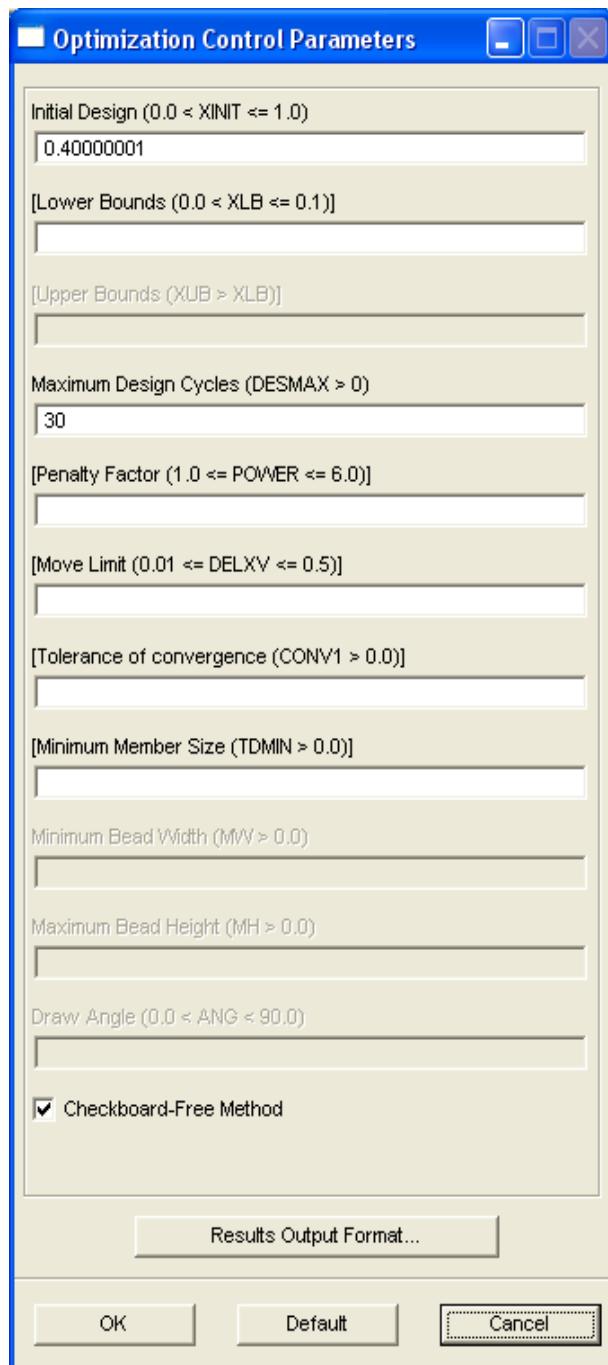


Figure 7-19 Optimization Parameters

Step 8: Creating and Selecting Subcase for Optimization

After the Optimization problem has been setup, the next requirement is the creation and selection of Subcase.

- a. Click **Analysis**.
- b. Select **Action**: Optimize; **Object**: Entire Model; **Method**: Analysis Deck in Analysis form
- c. Click the **Subcase Create** tab.
- d. In the Subcase Create form, select **Solution Type**: 101 Linear Static
- e. Select the created Load Case from the Available Subcases list.
- f. Click **Apply**, and **Cancel**.
- g. Click the **Subcase Select** tab in the Analysis Form.
- h. In the Subcase Select form, select the created subcase from Subcases Available.
- i. Select **Solution Type**: 101 Linear Static
- j. Click **OK**.
- k. Click **Apply** in the Analysis form to create the analysis deck for optimization.

This completes the Preprocess.



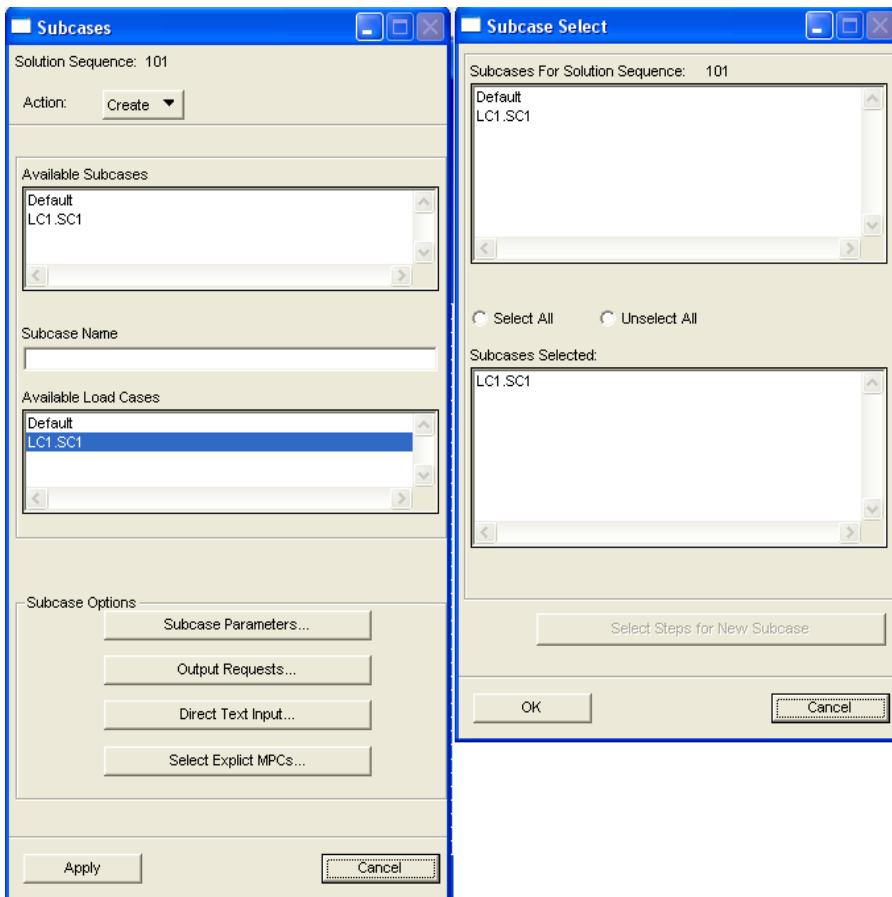


Figure 7-20 Creation and Selection of Subcase

Submit the analysis deck to MSC Nastran. (As indicated in [Initiating Topology Toptimization](#); this can be done from the GUI itself if Method is Full Run instead of Analysis Deck). The density distribution file (.des) will be created. This can be read by Patran and displayed as a density plot which is the topology design proposed by MSC Nastran. This is achieved by Patran postprocessing and the steps are now described.

Step 9: Reading jobname.des file into Patran

- Click **Tools**→**Design Study**→**Post Process**
- Select **Action**: Read Results.
- Click the **Select Results File** tab.
- Select the desired (.des) file through the browser.
- Click **Apply**.



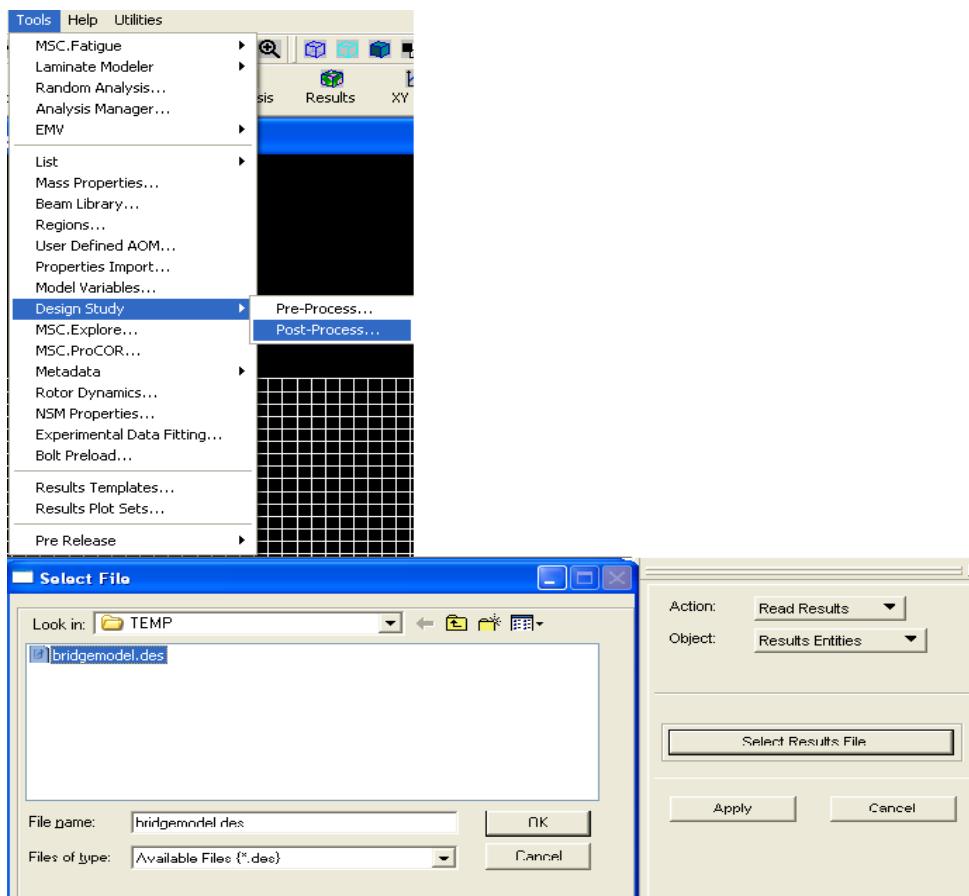


Figure 7-21 Reading Density Distribution File into Patran

Step 10: Display Fringe Plot

- Select **Action:** Display Results.
- Select last design cycle in the Select Result Case list.
- Input 0.4 as the Threshold.
- Select the **Fringe** check box.
- Click **Apply**.



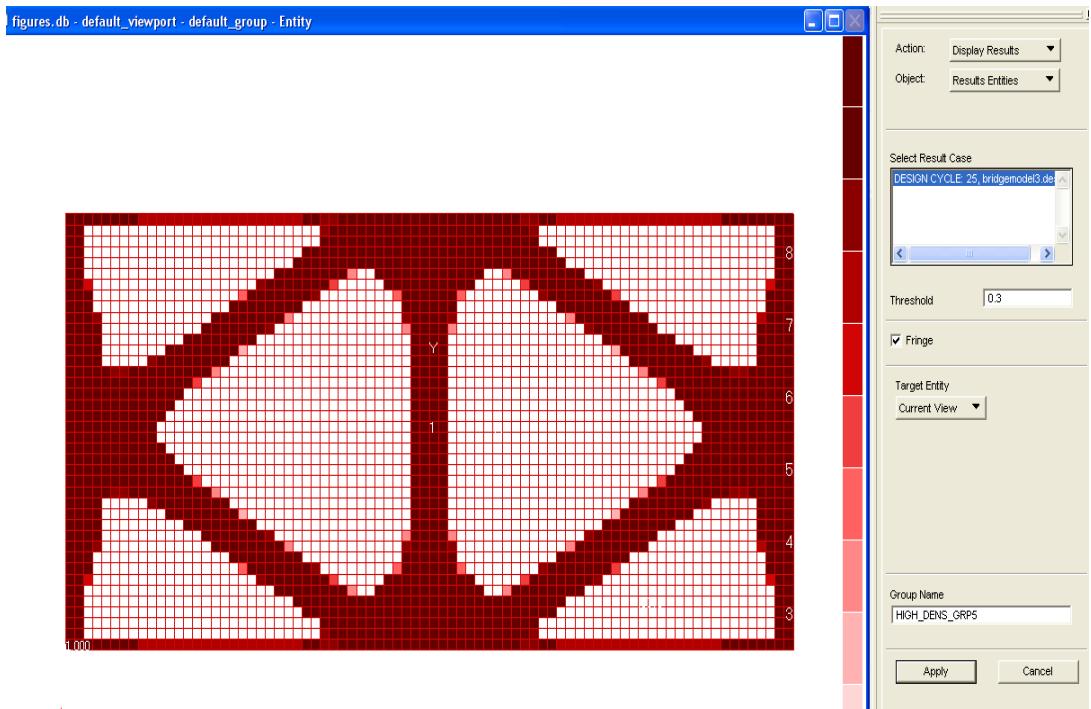


Figure 7-22 Display of Fringe Plot

Step 11: Smooth Results

- a. Select **Action**: FEM Smooth; **Method**: 2D
- b. Select **Remesh EIType**: Quad4
- c. Input 0.4 for Threshold.
- d. Make a window around the full design domain to input Select FEM to smooth.
- e. Click **Apply**.

A smoothed and remeshed topology design is displayed.



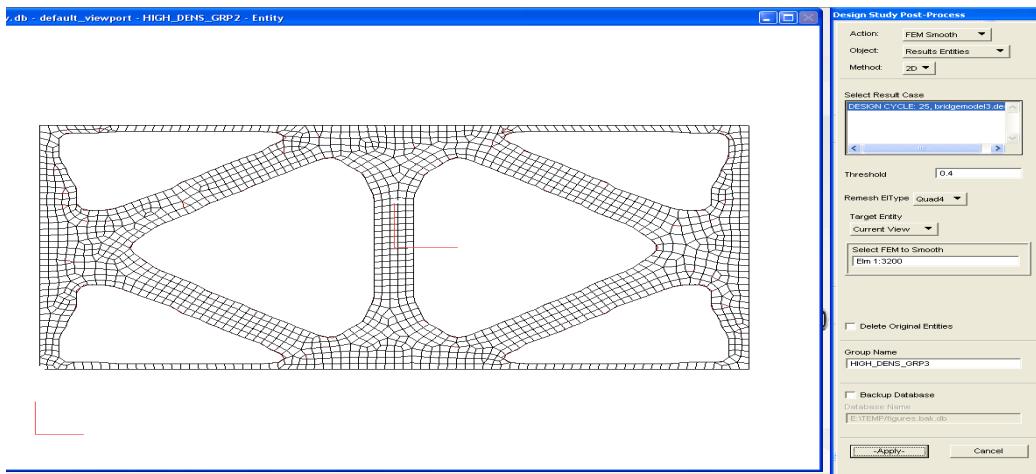


Figure 7-23 Smoothed and Remeshed Topology

Further work with this model requires that new geometry be created in a CAD system. For this purpose the new surfaces appearing in the proposed topology design are to be captured and then the geometric model can be exported as an IGES file. This process is now explained.

Step 12: Capturing new Surfaces appearing in the Proposed Topology Design

- Click **Geometry**.
- In the Geometry form, select **Action: Create**; **Object: Surface**; **Method: Mesh**.
- Make a window around the full design domain to create input for the Element List.
- Click **Apply**.



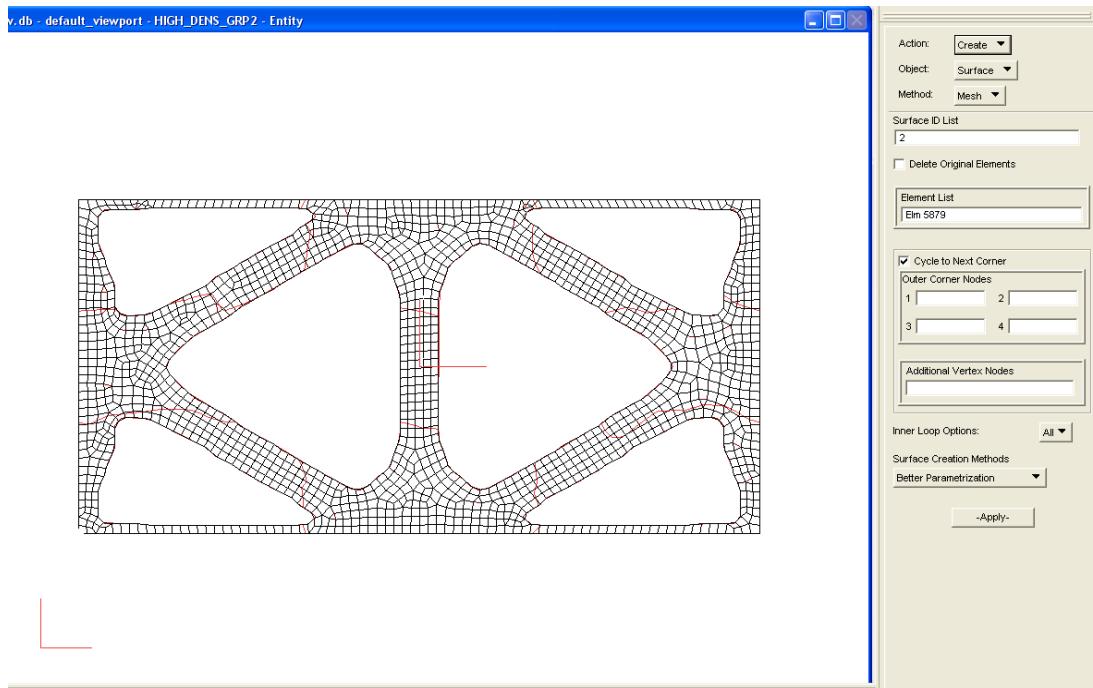


Figure 7-24 Creating new Surface

Step 13: Associate remaining Finite Elements in the Model with newly Created Surface

- a. Click **Element**.
- b. In the Element form, select **Action**: Associate; **Object**: Element; **Method**: Surface
- c. Uncheck **Auto Execute**.
- d. Make a window around the full domain to create input for the Element List.
- e. Click on the newly created surface as input to the Surface List.
- f. Click **Apply**.

Now the model has all the FE and geometric information and its association.



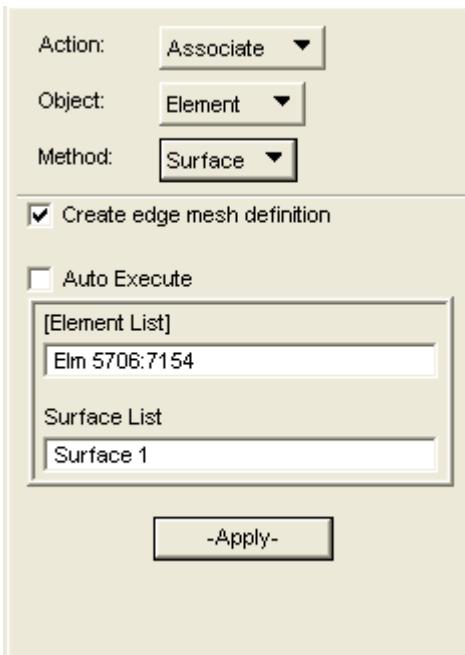


Figure 7-25 Associating FE with Surface

Step 14: Creation of IGES file

- a. Click **Preferences→Global**
- b. Input 0.6 for the Global Tolerance (relax the Global Tolerance).
- c. Click **Apply**.
- d. Click **File→ Export**
- e. In the Export form, select **Format: IGES**.
- f. Check Export through Parasolid box.
- g. Enter a File name.
- h. Click the **IGES Options** tab.
- i. In the form, select Surfaces as the Entity Types.
- j. Click **OK**.
- k. Click **Apply**.



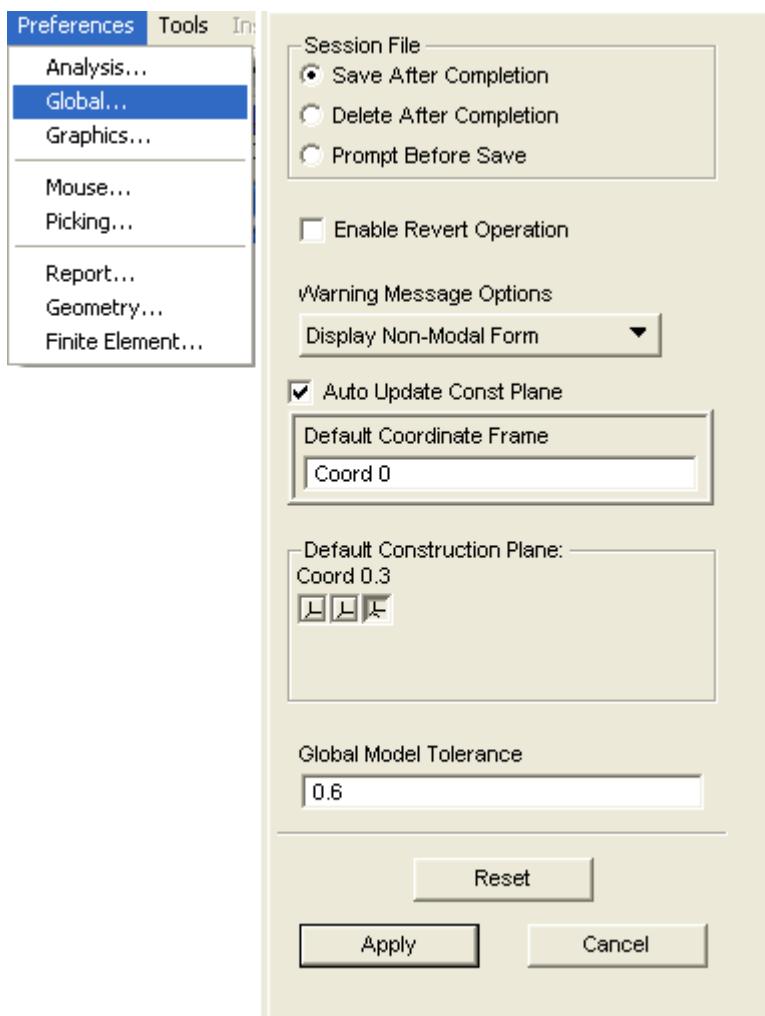


Figure 7-26 Relaxing Global Tolerance



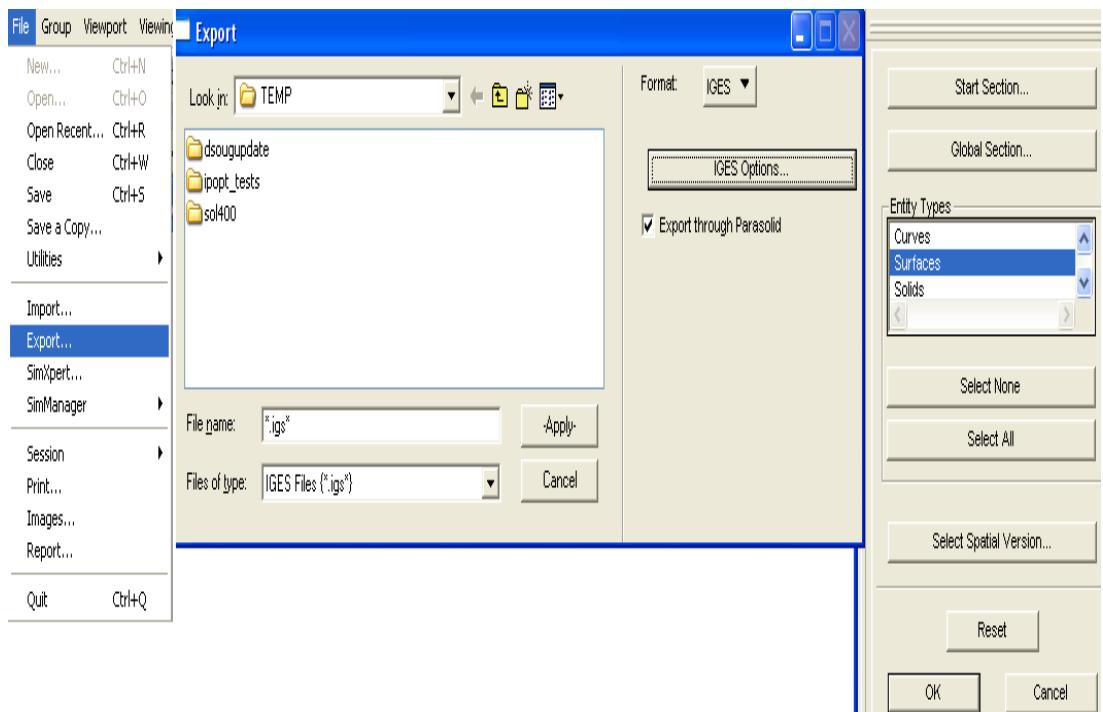


Figure 7-27 Exporting IGES file



MBB Beam with Variations

Summary

ATTRIBUTE	VALUE
Title	MBB Beam
Topology Optimization features	Compliance minimization Mass target Checkerboard free solution Minimum member size control Mirror symmetry constraints
Geometry	 Length = 6 and width = 2 Thickness = 0.01
Material	E = 2.05E+5 Pa, $\mu=0.3$
Analysis	Static analysis
Boundary conditions	Supported on rollers at one point and fixed support at another point
Applied loads	A concentrated force = 100.0 N
Element types	4 node linear QUAD elements
Topology result	Material distribution

Introduction

An MBB beam example (a half model shown in [Figure 7-28](#)) is used to demonstrate (a) basic MSC Nastran Topology Optimization capabilities without manufacturing constraints (topex3.dat in TPL), (b) minimum member size control (topex3a.dat in TPL), and (c) mirror symmetry constraints (topex3b.dat in TPL). The structural compliance (i.e., total strain energy) is minimized with a mass target 0.5 (i.e., 50% material savings). The loading and boundary conditions are shown in [Figure 7-28](#). The structure is modeled with 4800 CQUAD4 elements.



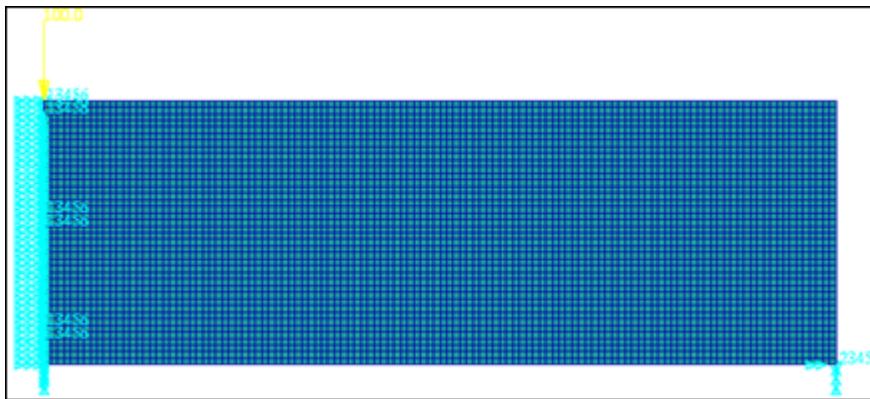


Figure 7-28 MBB Beam

Solution Requirements

This MBB example is widely used by academic and industrial researchers for Topology Optimization validation.

Design Model Description

Objective: Minimize compliance

Topology design region: PSHELL

Constraints: Mass target = 0.5 (i.e. mass savings 50%)

- a. Minimum member size control and/or
- b. Mirror symmetry constraints

These solutions demonstrate:

- A distinct design can be obtained by MSC Nastran Topology Optimization with checkerboard free algorithm (as default)
- The minimum member size is mainly used to control the size of members in topology optimal designs. Preventing thin members enhances the simplicity of the design and hence its manufacturability. Minimum member size is more like quality control than quantity control.
- By using symmetry constraints in Topology Optimization, a symmetric design can be obtained regardless of the boundary conditions or loads.

Optimization Solution

Basic Compliance Minimization

The input data for this example related to Topology Optimization model is given in [Listing 7-2](#). A Bulk Data entry TOPVAR =1 is used to define a topological design region. XINIT=0.5 on the TOPVAR entry



matches the mass target constraint so that the initial design is feasible. The rest of the values on the TOPVAR entry are default values that are recommended for general Topology Optimization applications. Type one design responses DRESP1 = 1 and 2 identify compliance and fractional mass respectively. DCONSTR= 1 specifies the mass target. DESOBJ=1 in Case Control Command selects DRESP1=1 entry to be used as a design objective (minimization as default) and DESGLB selects the design constraint DCONSTR= 1 to be applied in this Topology Optimization task.

Listing 7-2 Input File for MBB Beam

```

DESOBJ = 1
DESGLB = 1
SUBCASE 1
$ Subcase name : Default
    SUBTITLE=Default
    SPC = 2
    LOAD = 2
    ANALYSIS = STATICS
BEGIN BULK
DCONSTR 1      2          .5
TOPVAR,   1     ,      Tshell,      Pshell, , , , 1
DRESP1   1     COMPL     COMP
DRESP1   2     FRMASS    FRMASS

```

Figure 7-29 shows the topology optimized result that is smoothed and remeshed by using Patran. This optimal design is very clear without any checkerboard effect. It is noticed that there are some small members.

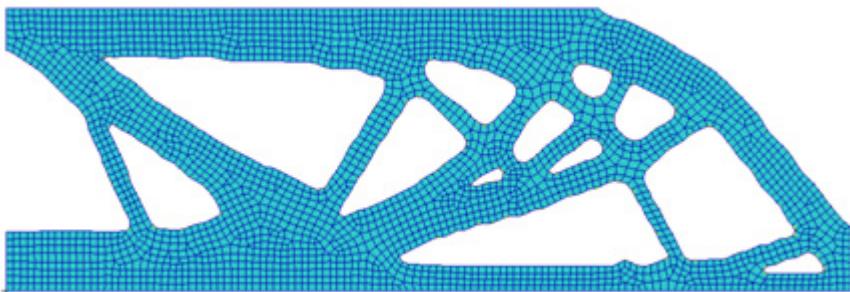


Figure 7-29 MBB Beam Topology Design

Minimum Member Size Control

The MBB beam (shown in [Figure 7-28](#)) is used again to demonstrate the minimum member size control capability.

The input data for this example related to Topology Optimization with "minimum member size" is given in [Listing 7-3](#). The minimum member size value is defined by the parameter TDMIN = 0.5 on the DOPTPRM entry and corresponds to the length of 10 elements.

Listing 7-3 Input File for MBB Beam with Minimum Member Size

```

DESOBJ = 1
DESGLB = 1

```



```

SUBCASE 1
$ Subcase name : Default
  SUBTITLE=Default
  SPC = 2
  LOAD = 2
  ANALYSIS = STATICS
BEGIN BULK
DOPTPRM, TDMIN, 0.5
DOCONSTR 1      2           .5
TCPVAR,        1 ,   Tshell,    Pshell, , , , 1
DRESP1   1      COMPL   COMP
DRESP1   2      FRMASS  FRMASS

```

The [Figure 7-30](#) shows the topology optimized result with "minimum member size" TDMIN=0.5. Compared to the design shown in [Figure 7-29](#), this design with "minimum member size" is obviously much simpler and there are no tiny members.

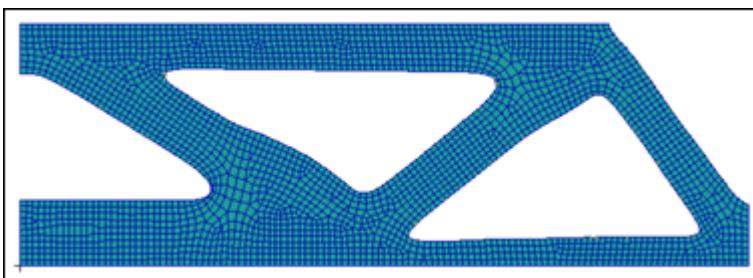


Figure 7-30 MBB Beam Topology Design with "Minimum Member Size"

Mirror Symmetric Constraints

Since the loads applied on the MBB beam are not symmetric, the topology optimized designs [Figure 7-29](#) and [Figure 7-30](#) are not symmetric. The MBB beam is employed again to demonstrate the mirror symmetric constraint capability that enforces the design to be symmetric about a given plane.

To apply symmetric constraints on designed properties, users need to create a reference coordinate system using a rectangular coordinate system CORD1R or CORD2R. In this example, grid 10001 (location x=3, y=1, and z=0) is defined as the origin. Grid 10002 (x=3, y=1, and z=1) lies on the z-axis, and grid 1003 (x=4, y=1, and z=0) lies in the x-z plane. CORD1R CID=1 defines a reference coordinate system. A continuation line "SYM" enforces the property PSHELL=1 to be symmetric about the planes YZ and ZX in the reference coordinate system CID=1. In addition, a minimum member size TDMIN=0.15 is applied. The input data for this example is given in [Listing 7-4](#).

Listing 7-4 Input File for MBB Beam with Mirror Symmetry Constraints

```

DESOBJ = 1
DESGLB = 1
SUBCASE 1
  SUBTITLE=Default
  SPC = 2
  LOAD = 2
  ANALYSIS = STATICS
BEGIN BULK
CORD1R  1       10001   10002   10003
GRID     10001   3.        1.        0.0

```



```
GRID      10002          3.      1.      1.0
GRID      10003          4.      1.      0.0
TOPVAR,  1      , Tshell, Pshell, , , , 1
        , SYM, ' 1, YZ, , ZX
        , TDMIN, 0.15
DRESP1   1      COMPL  COMP
DRESP1   2      FRMASS FRMASS
DCONSTR  1      2           .5
```

The [Figure 7-31](#) shows the topology optimal result with symmetric constraints and minimum member size=0.15, with sym= yz and zx, and xinit field blank.

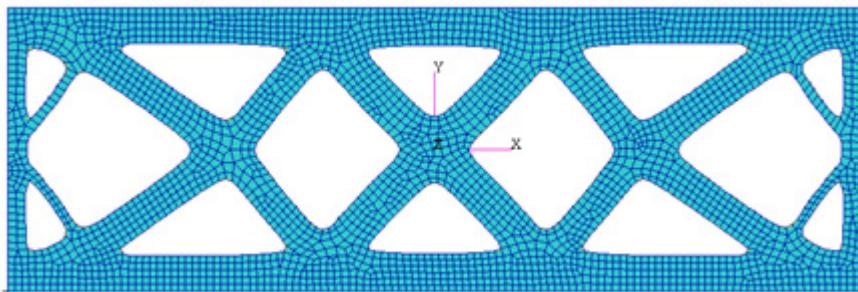
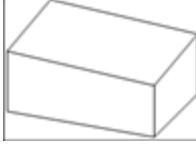


Figure 7-31 MBB Beam with Symmetric Constraints and Minimum Member Size



Torsion Beam with Variations

Summary

ATTRIBUTE	VALUE
Title	A Torsion Beam
Topology Optimization features	Compliance minimization Mass target Casting constraints Extrusion constraints Mirror symmetry constraints
Geometry	 Length = 16 and width = 4 height = 4
Material	E = 2.1E+5 Pa, $\mu=0.3$, and RHO=1.0
Analysis	Static analysis
Boundary conditions	Fixed support at one end
Applied loads	A pair of twisting forces = 1000.0 N
Element types	8 node HEXA
Topology result	Material distribution

Introduction

A torsion beam is used here to demonstrate extrusion (topex5.dat in TPL), one-die casting constraint (topex5a.dat in TPL) and two-die casting constraints (topex5b.dat in TPL). [Figure 7-32](#) shows the FEM model of the torsion beam. A pair of twisting forces is applied on one end while the other end is fixed. 2048 CHEXA elements are used for this model. The objective is to minimize the structural compliance with mass target of 0.3 (i.e., 70% material savings).



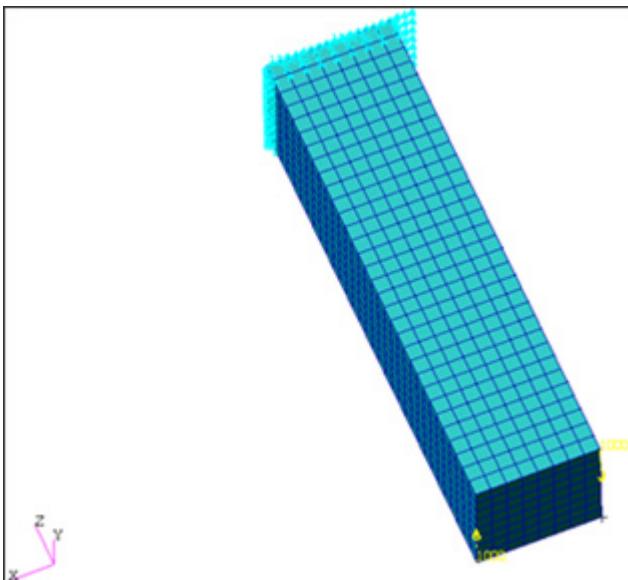


Figure 7-32 Torsion Beam

Solution Requirements

This torsion beam is utilized to show MSC Nastran Topology Optimization extrusion and casting constraint capabilities.

Design Model Description

Objective: Minimize compliance

Topology design region: PSOLID

Constraints: Mass target = 0.3 (i.e., mass savings 70%)

- (a) Extrusion constraints or
- (b) Casting constraints with one or two dies

These solutions demonstrate:

- By using extrusion constraints in Topology Optimization, a constant cross-section design along the given extrusion direction can be obtained regardless of the boundary conditions or loads.
- The use of casting constraints can prevent hollow profiles in Topology Optimization so that a die can slide in a given direction. One or two dies options are available for selection.
- Some combined manufacturing constraints are allowed in Topology Optimization to achieve the design goal.



Optimization Solution

Extrusion Constraints With One Die

It is often seen that some topology optimized designs contain cavities that are not achievable or require a high cost manufacturing process. For example, the result from the torsion beam without manufacturing constraints is shown in [Figure 7-33](#). Clearly, this topology design proposal is not achievable by casting.

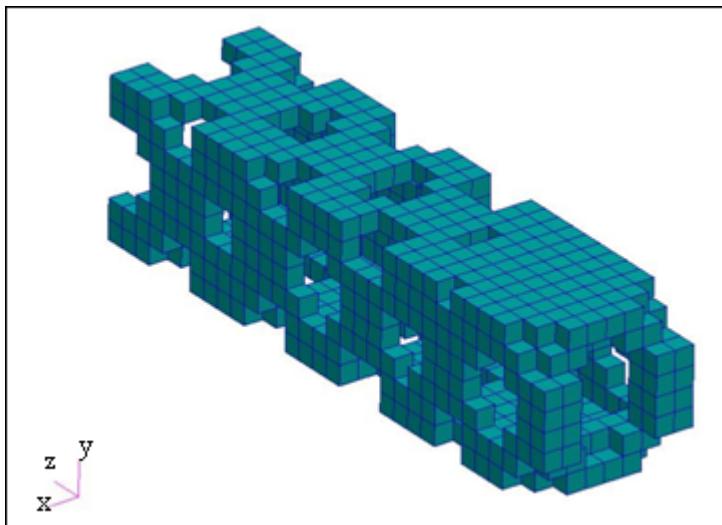


Figure 7-33 Torsion Beam without Manufacturing Constraints

The extrusion constraints enforce a constant cross-section design along the given extrusion direction. The input data related to imposing an extrusion constraint along the z-axis in the basic coordinate system (as the default option) is given in [Listing 7-5](#).

Listing 7-5 Input File for Torsion Beam with Extrusion

```
DESOBJ = 1
DESGLB = 1
SUBCASE 1
    SUBTITLE=Default
    ANALYSIS = STATICS
    SPC = 2
    LOAD = 2
BEGIN BULK
DRESP1 2      Frmass   FRMASS
DRESP1 1      COMPL    COMP
DCONSTR 1      2          .3
TOPVAR, 1      ,      TSOLID,      PSOLID, .3, , , , 1
        EXT  ,      ,      Z
PSOLID 1      1          0
```

The [Figure 7-34](#) shows the topology optimized result with extrusion constraints. It is obvious that the design has a constant cross-section along the z-axis.



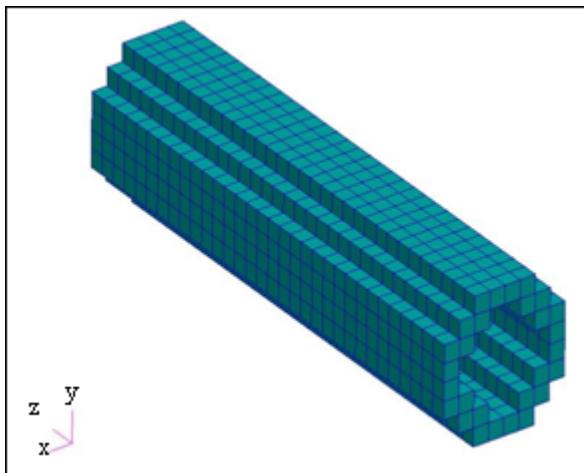


Figure 7-34 Torsion Beam with Extrusion Constraints in Z-Axis

Casting Constraints with One Die

A torsion beam (shown in Figure 7-32) is used here to demonstrate the combination of one die casting manufacturability constraints and mirror symmetric constraints. (Figure 7-34)

The casting constraints with one die option enforce the material to be added to the region by "filling up" in the given draw direction from the bottom (or, stated another way, that voids extend from the top surface and do not reappear in the die direction). To apply casting constraints and symmetric constraints on designed properties, a reference coordinate system CID=1 is defined by using a rectangular coordinate system CORD1R. A "CAST" continuation line defines casting constraints in the Y direction and one die is a default option. Another "SYM" continuation line defines symmetric constraints about the YZ plane. The input data related to the Topology Optimization model is given in Listing 7-6.

Listing 7-6 Input File for Torsion with One Die

```

DESOBJ = 1
DESGLB = 1
SUBCASE 1
    SUBTITLE=Default
    ANALYSIS = STATICS
    SPC = 2
    LOAD = 2
BEGIN BULK
    DRESP1 2      Frmass   FRMASS
    DRESP1 1      COMPL    COMP
    DCONSTR 1     2          .3
    CORD1R 1      5          167     7
    PSOLID 1      1          0
    TOPVAR, 1,     TSOLID,   PSOLID,   .3, , , 4.0,   1
                           ,       CAST,     1, , Y
                           ,       SYM,      1, , YZ, , YES

```



The [Figure 7-35](#) shows the topology optimized result with one die casting constraint. It is observed that the design material is added by "filling up" in the Y direction from the bottom. In addition, the design is symmetric about the YZ plane in the reference coordinate system CID=1.

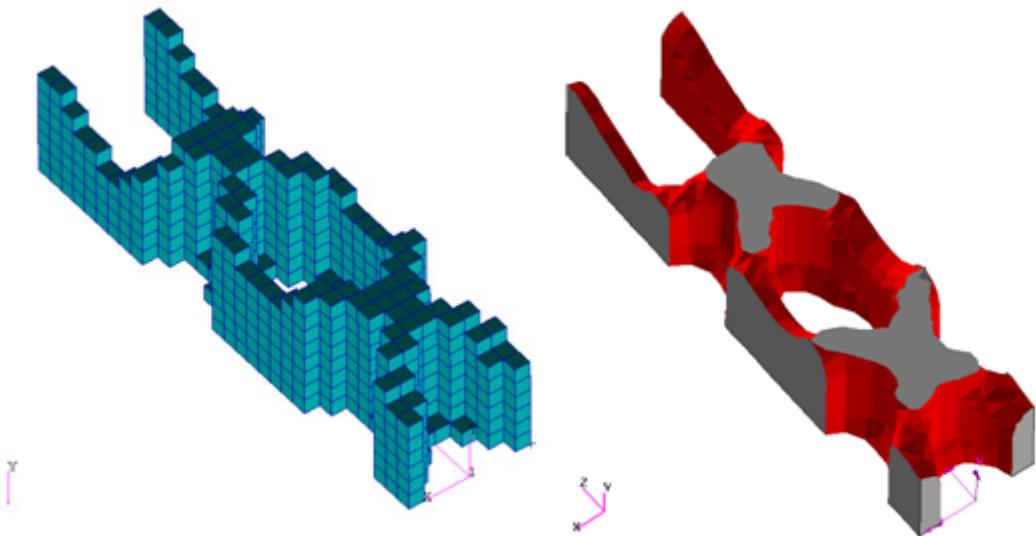


Figure 7-35 Torsion Beam with One Die Casting Constraints in Y Direction

Casting Constraints with Two Dies

The torsion beam (shown in [Figure 7-32](#)) is again used here to demonstrate two die casting manufacturability constraints.

The input for two die casting constraints is similar to the one die option in the previous example the difference is that 2 is selected for the DIE field on the TOPVAR entry. The input data related to imposing two die casting constraints is given in [Listing 7-7](#).

Listing 7-7 Input File for Torsion with Two Dies

```
DESOBJ = 1
DESGLB = 1
SUBCASE 1
$ Subcase name : Default
SUBTITLE=Default
ANALYSIS = STATICS
SPC = 2
LOAD = 2
$ Direct Text Input for this Subcase
BEGIN BULK
DRESP1 2      Frmass   FRMASS
DRESP1 1      COMPL    COMP
DCONSTR 1     2          .3
CORD1R  1     5          167     7
PSOLID   1     1          0
TOPVAR,      1 ,      TSOLID,      PSOLID,      .3
,           ,      CAST,        1 ,      .3
,           ,      Y,      2, YES
```



, SYM , 1 , YZ

The [Figure 7-36](#) shows the topology optimized result with two die casting constraints. It is observed that the design material grows from the splitting plane in opposite directions along the y-axis specified in the reference coordinate system CID=1. The splitting plane is determined by optimization and in this case corresponds to the y-z plane.

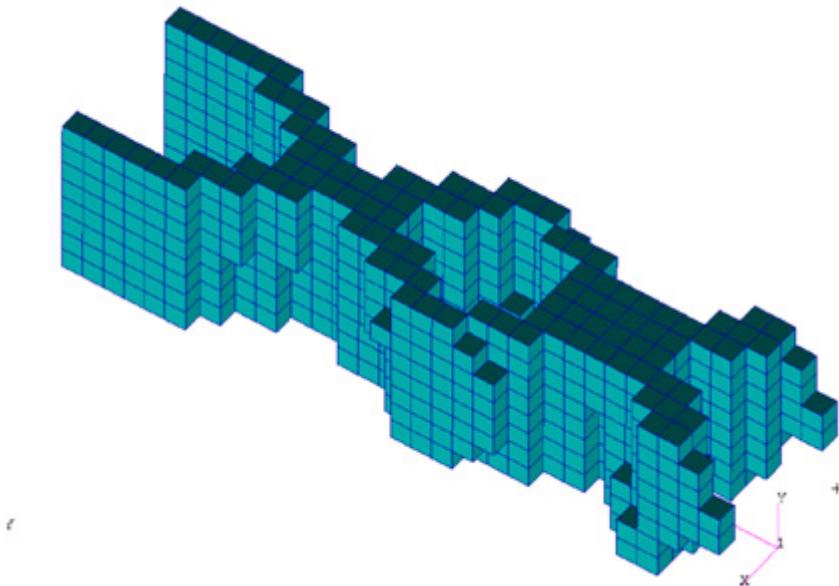
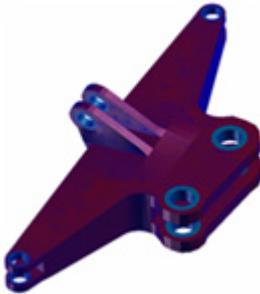


Figure 7-36 Torsion Beam with Two Die Casting Constraints in Y-Axis



Engine Mount

Summary

ATTRIBUTE	VALUE
Title	Engine Mount
Topology Optimization features	Averaged compliance minimization Multiple TOPVAR entries Multiple load cases Displacement constraints
Geometry	
Material	$E = 2.05E+5 \text{ Pa}$, $\mu=0.3$
Analysis	Static analysis
Boundary conditions	Fixed at three points
Applied loads	14 load cases (forces)
Element types	HEXA, PENTA, TETRA, and RBE3
Topology result	Material distribution

Introduction

The main goal of this engine mount example is to minimize the compliance of the engine-front-mount-beam (shown in [Figure 7-37](#)) with mass target 0.3 (material savings 70%) and displacements within a range (-0.6, 0.6) at 5 selected grids. The analysis model has 14 load cases. The finite element model is shown in [Figure 7-38](#). There are 62306 HEXA elements, 703 PENTA elements, 31 TETRA elements, and 5 RBE3 elements. (Refer topoug2.dat in TPL).



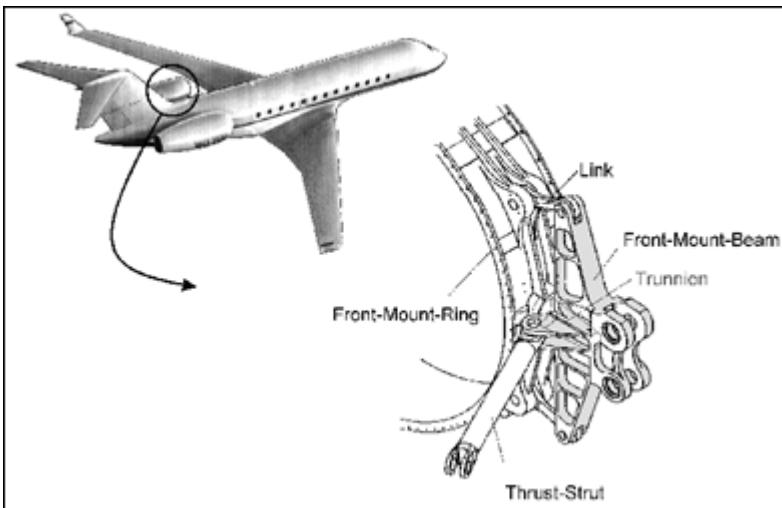


Figure 7-37 Front-Mount-Beam

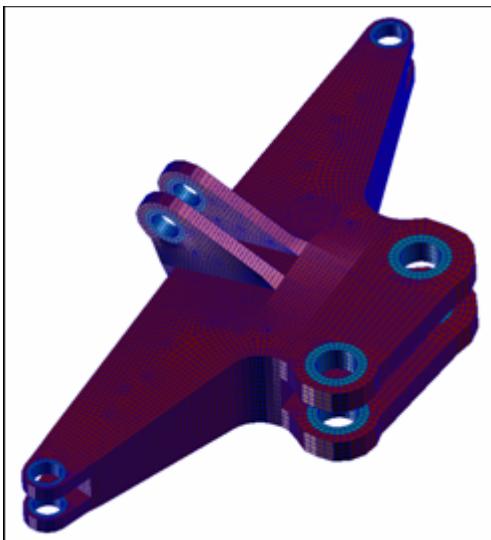


Figure 7-38 Front-Mount-Beam FE Model

The objective of this example is to illustrate the step-by-step procedure to (a) setup a Topology Optimization task with displacement constraints and multiple load cases in Patran using General Topology Optimization interface and (b) use Patran Insight to get the iso-surfaces.

Solution Requirements

Design Model Description



Objective: Minimize averaged compliance
 Topology design region: PSOLID = 1, 2, 3, 8, 9 and 10

Constraints: Mass target = 0.3 (i.e., mass savings 70%) Displacements at grid points 76095, 76096, 76419, 76420, and 76421 for all 14 load cases within the range (-6.0, 6.0)

This solution demonstrates:

- The averaged compliance can be used for Topology Optimization problems with multiple load cases to achieve an efficient design concept.
- Multiple topological design parts are allowed
- Displacement constraints can be readily handled in Topology Optimization.

Optimization Solution

Topology Optimization Input through Patran

Create a new database in Patran and Import the model file as an MSC Nastran input file. Next create 14 Load cases. The latter two steps have already been explained in Patran Tutorial 1. The following table gives the load and displacement sets belonging to each load case.

Load Case	Displacement ids	Force ids
1	Displ_spc.1, Displ_spc.1.cid3	Force_force1.cid4, 5, 6, 7, 8, 9
2	Displ_spc.1, Displ_spc.1.cid3	Force_force2.cid4, 5, 6, 7, 8, 9
3	Displ_spc.1, Displ_spc.1.cid3	Force_force3.cid4, 5, 6, 7, 8, 9
4	Displ_spc.1, Displ_spc.1.cid3	Force_force4.cid4, 5, 6, 7, 8, 9
5	Displ_spc.1, Displ_spc.1.cid3	Force_force5.cid4, 5, 6, 7, 8, 9
6	Displ_spc.1, Displ_spc.1.cid3	Force_force6.cid4, 5, 6, 7, 8, 9
7	Displ_spc.1, Displ_spc.1.cid3	Force_force7.cid4, 5, 6, 7, 8, 9
8	Displ_spc.1, Displ_spc.1.cid3	Force_force8.cid4, 5, 6, 7, 8, 9
9	Displ_spc.1, Displ_spc.1.cid3	Force_force9.cid4, 5, 6, 7, 8, 9
10	Displ_spc.1, Displ_spc.1.cid3	Force_force10.cid4, 5, 6, 7, 8, 9
11	Displ_spc.1, Displ_spc.1.cid3	Force_force11.cid4, 5, 6, 7, 8, 9
12	Displ_spc.1, Displ_spc.1.cid3	Force_force12.cid4, 5, 6, 7, 8, 9
13	Displ_spc.1, Displ_spc.1.cid3	Force_force13.cid4, 5, 6, 7, 8, 9
14	Displ_spc.1, Displ_spc.1.cid3	Force_force14.cid4, 5, 6, 7, 8, 9

Step 1: Initiating the General Topology Optimization interface.

This process enables the input of all Topology Optimization parameters for setup of the optimization job.

Click on Tools→ Design Study→ Pre Process



The form that appears enables creation of TOPVAR variables, objective function, constraint functions, constraint sets and a Design Study.

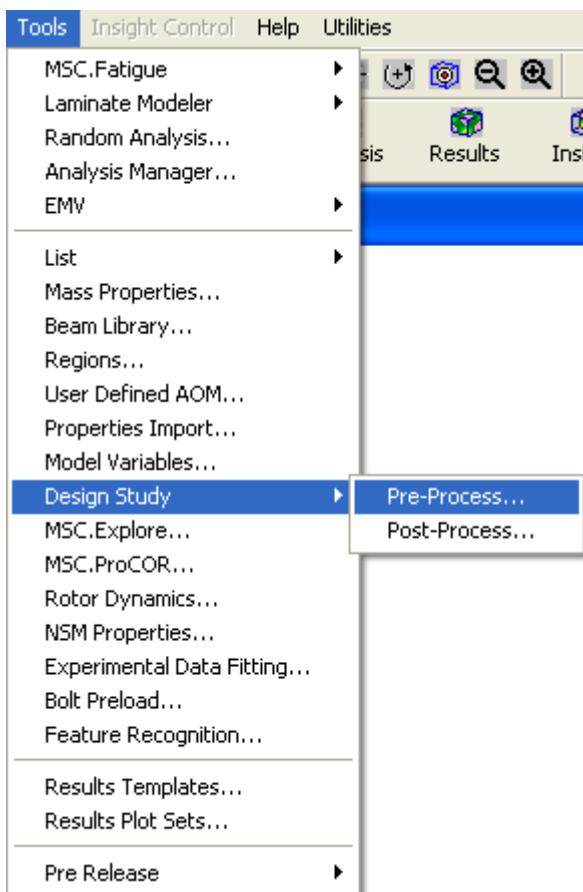


Figure 7-39 General Topology Optimization

Step 2: Creation of initial TOPVAR variable

- a. Select **Action**: Create; **Object**: Design Variable; **Type**: Topology
- b. Select Dimension 3-D; Type: Solid
- c. Select psolid.1 in Select Property Set list. A variable name tv_psolid.1 will automatically appear in the Variable Name box.
- d. Click **Apply**.



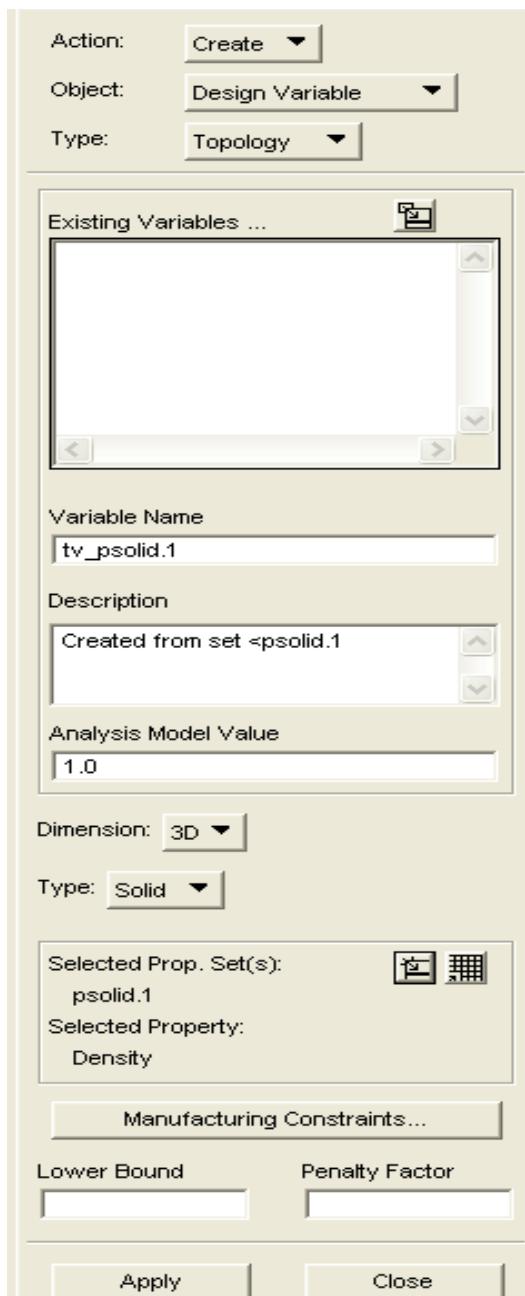


Figure 7-40 Creation of TOPVAR



Similarly, create 5 more TOPVAR variables. The table below gives the designable property id for each of the created TOPVAR variables.

TOPVAR ID	PSOLID ID
1 (tv_psolid.1)	1 (psolid.1)
2 (tv_psolid.2)	2 (psolid.2)
3 (tv_psolid.3)	3 (psolid.3)
4 (tv_psolid.7)	7 (psolid.7)
6 (tv_psolid.8)	8 (psolid.8)
7 (tv_psolid.9)	9 (psolid.9)

Step 3: Creation of Objective

The objective is minimization of the average compliance of all the 14 load cases.

- a. Select **Action**: Create; **Object**: Objective; **Solution**: Global; **Response**: Average Compliance.
- b. Select **Min/Max**: minimize.
- c. Input the Objective Name (avgcomp).
- d. Click **Apply**.

Step 4: Creation of Global Constraint

- a. There is a global constraint of 30% mass fraction.
- b. Selecting **Action**: Create; **Object**: Constraint; **Solution**: Global; **Response**: Fractional Mass.
- c. Input 0.3 as the Upper Bound.
- d. Input mass as the Constraint Name.
- e. Click **Apply**.





Figure 7-41 Creating Objective



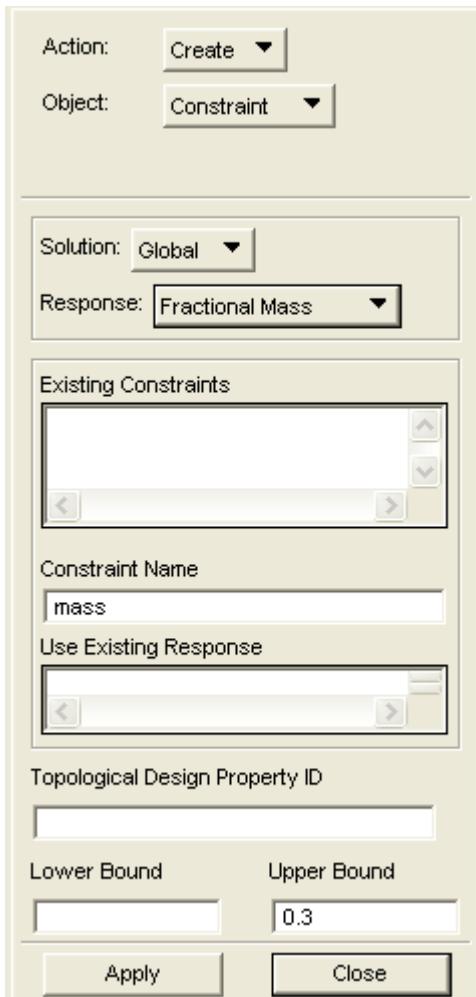


Figure 7-42 Creating Mass Constraint

In addition to the fractional mass global constraint, there are displacement constraints on 5 grid points under each of the 14 load cases. The total number of constraints is 70 grouped into 14 constraint sets of 5 constraints each. Now the steps to create these constraints and constraint sets are explained.

Step 5: Creation of Displacement Constraints

- Select **Action**: Create; **Object**: Constraint; **Solution**: Linear Static; **Response**: Displacement
- Input loadcase1grid1 for Constraint Name.
- Input Node 76095 for Select Node.
- Select -6 for Lower Bound and 6 for Upper Bound.
- Check Magnitude radio button in Displacement Component section.



f. Click **Apply**.

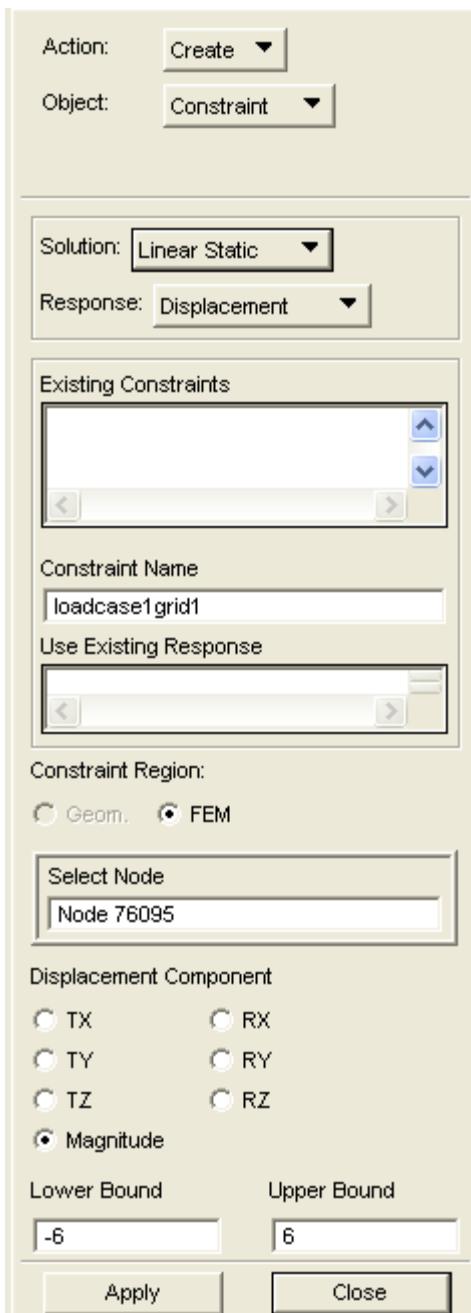


Figure 7-43 Creating Displacement Constraint



Similarly, create 69 more constraints. The following table gives the load case and grid number for each of the constraints. Each of these displacement responses is constrained to lie between -6 and 6.

Load Case	Grid 76095	Grid 76096	Grid 76419	Grid 76420	Grid 76421
1	loadcase1grid1	loadcase1grid2	loadcase1grid3	loadcase1grid4	loadcase1grid5
2	loadcase2grid1	loadcase2grid2	loadcase2grid3	loadcase2grid4	loadcase2grid5
3	loadcase3grid1	loadcase3grid2	loadcase3grid3	loadcase3grid4	loadcase3grid5
4	loadcase4grid1	loadcase4grid2	loadcase4grid3	loadcase4grid4	loadcase4grid5
5	loadcase5grid1	loadcase5grid2	loadcase5grid3	loadcase5grid4	loadcase5grid5
6	loadcase6grid1	loadcase6grid2	loadcase6grid3	loadcase6grid4	loadcase6grid5
7	loadcase7grid1	loadcase7grid2	loadcase7grid3	loadcase7grid4	loadcase7grid5
8	loadcase8grid1	loadcase8grid2	loadcase8grid3	loadcase8grid4	loadcase8grid5
9	loadcase9grid1	loadcase9grid2	loadcase9grid3	loadcase9grid4	loadcase9grid5
10	loadcase10grid1	loadcase10grid2	loadcase10grid3	loadcase10grid4	loadcase10grid3
11	loadcase11grid1	loadcase11grid2	loadcase11grid3	loadcase11grid4	loadcase11grid3
12	loadcase12grid1	loadcase12grid2	loadcase12grid3	loadcase12grid4	loadcase12grid3
13	loadcase13grid1	loadcase13grid2	loadcase13grid3	loadcase13grid4	loadcase13grid3
14	loadcase14grid1	loadcase14grid2	loadcase14grid3	loadcase14grid4	loadcase14grid3

Each row of constraints in the above table is combined into a constraint set. Thus there are 14 constraint sets. It may be noted that in this example the same 5 displacement constraints apply to all the 14 load cases and therefore only 5 constraints would have sufficed.

Step 6: Creation of Constraint Sets

- Selecting **Action**: Create; **Object**: Constraint, **Solution**: Linear Static
- Enter a Constraint Set Name.
- Selecting the appropriate constraints from the Constraints to be included list.
- Click **Apply**.



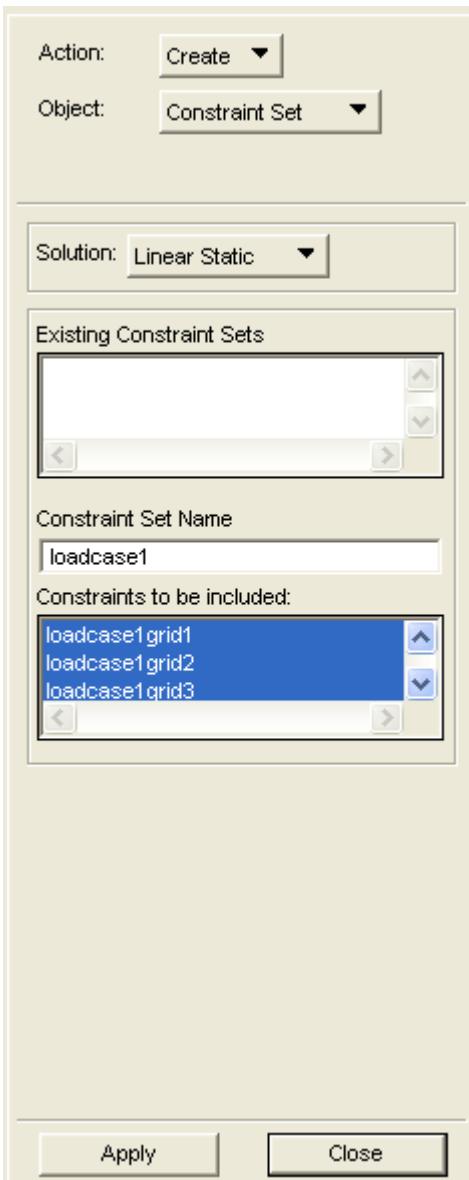


Figure 7-44 Creation of Constraint Set

Step 7: Creation of Design Study

Finally, a Design Study is created by selecting the created Design Variables, Objective, Constraints and Constraint Sets.

- Select **Action**: Create; **Object**: Design Study and enter a Design Study Name.



- b. One by one, click all the tabs in the Design Study Setup section and select all the listed quantities in the forms (Select Design Variables, Select Objective, Select Constraints and Select Constraint Sets).
- c. Input 0.3 for the Initial Design Value

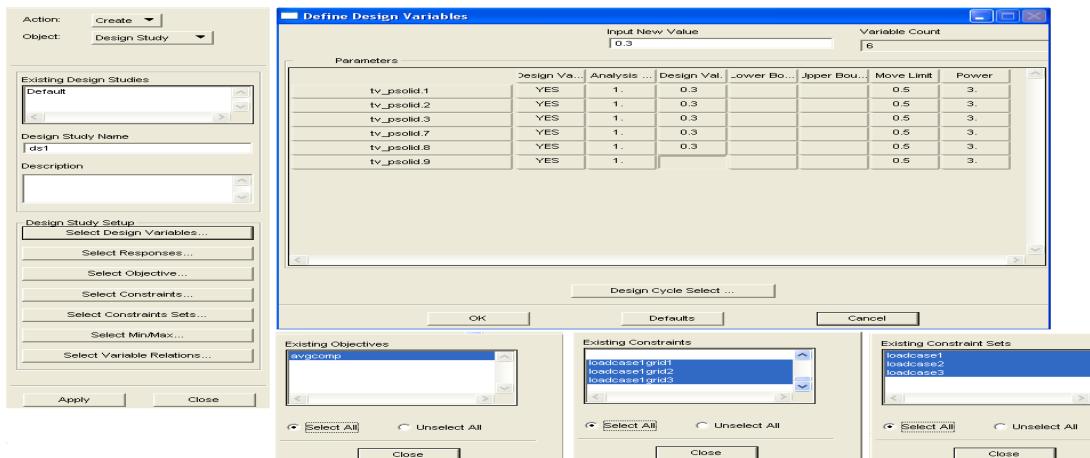


Figure 7-45 Creation of Design Study

Step 8: Creation of an Analysis Deck

- a. Click **Analysis**.
- b. In the Analysis form, select **Action**: Optimize; **Object**: Entire Model; **Method**: Analysis Deck.
- c. Click the **Design Study Select** tab and select ds1 (name of created design study) from the Existing Design Studies form.
- d. Click the **Global Obj./Constr. Select** tab and select avgcomp in Select an existing Global Objective list and mass in Select Existing Global Constraint(s) list.



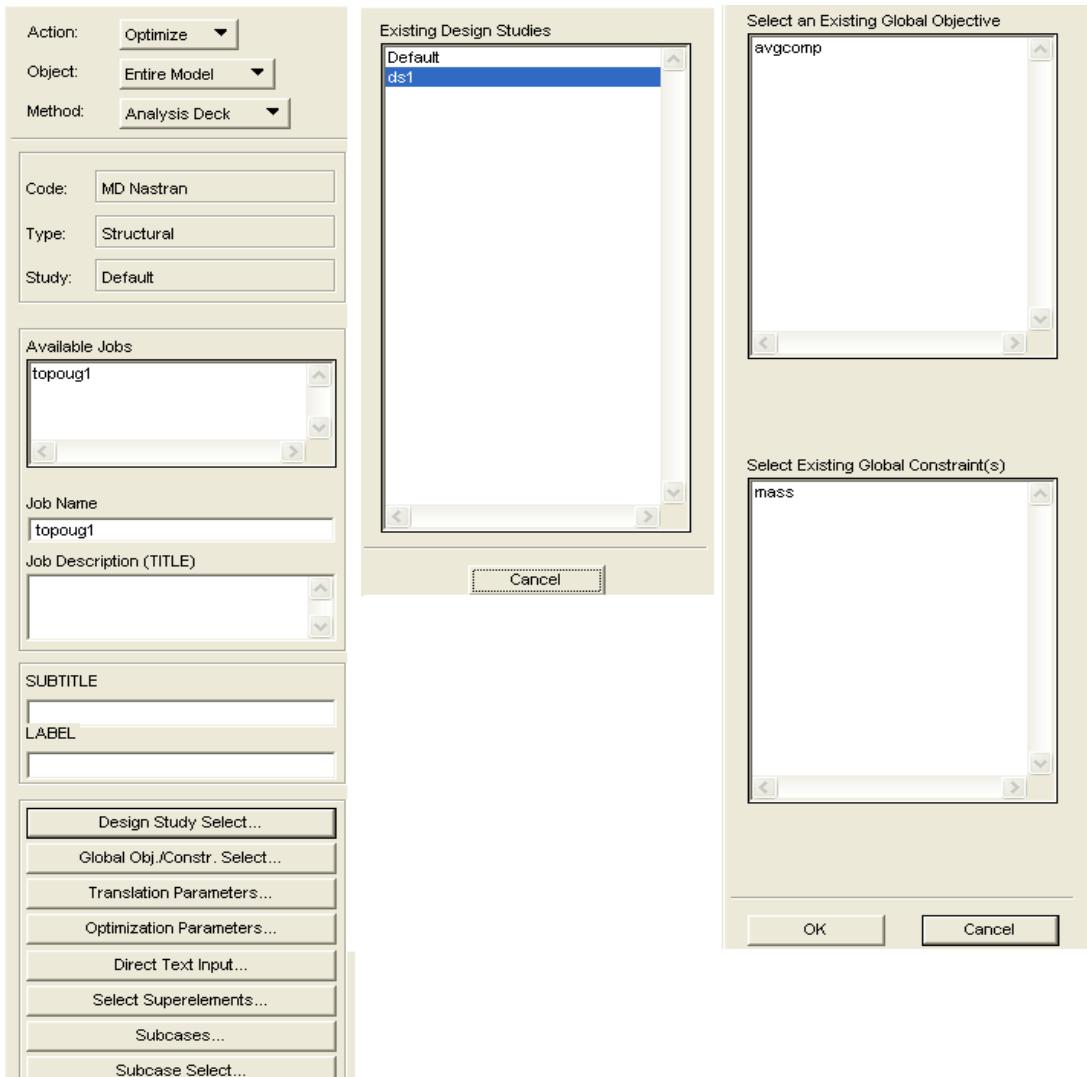


Figure 7-46 Creation of Analysis Deck

Step 9: Setting up of Optimization Parameters

- Click the **Optimization Parameters** tab.
- Select the Print Objective and Design Variables, Print Properties, Print all Constraints and Print all Responses check boxes.



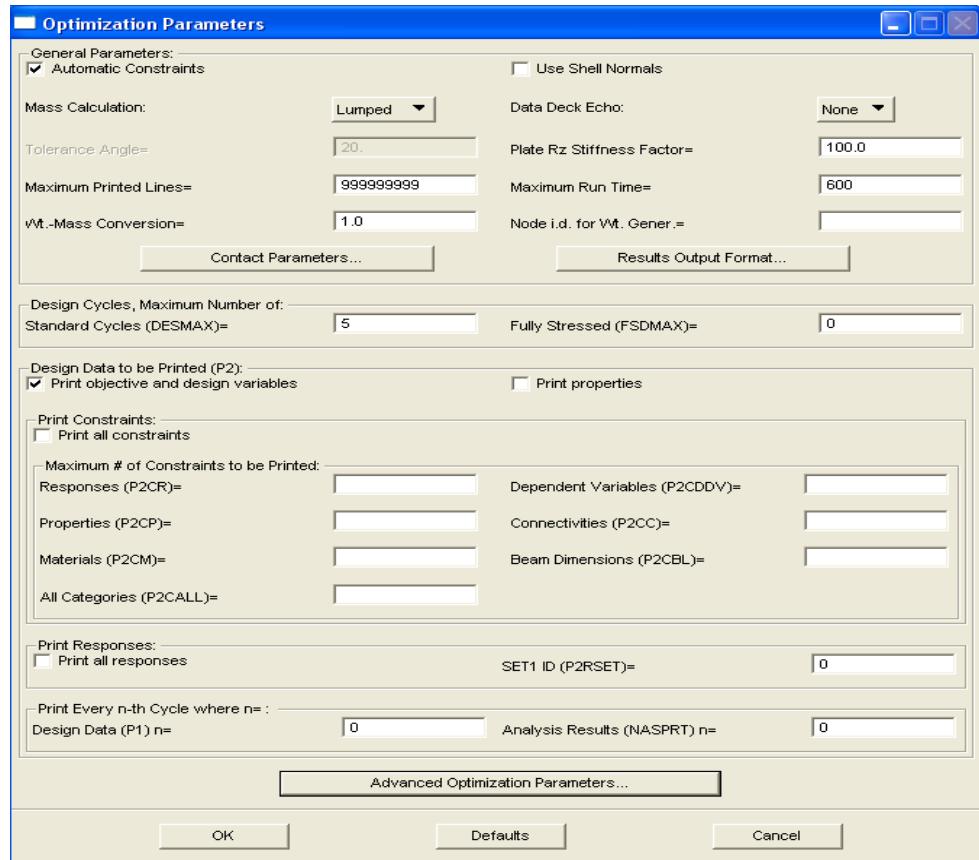


Figure 7-47 Setting up Optimization Parameters

Step 10: Creation and Selection of Subcases

Subcases will be created and the appropriate load cases and constraint sets will be associated with the subcases. Then the created subcases will be selected for the current job.

- Click the **Subcase** tab.
- In the Subcase Create form, select Solution Type: 101 LINEAR STATIC.
- Enter Subcase Name as subcase1.
- Click the **Select Constraints/Objective** tab.
- In the form, select loadcase1 from the Select Existing Constraint Sets list.
- Click **OK**.
- Click **Apply**.

Similarly create 13 more subcases by associating the remaining 13 load cases in sequence.

- Click the **Subcase Select** tab.



- i. Select Subcases 1 to 14 from the Subcases Available list.
- j. Click **OK**.
- k. Click **Apply** to generate the Analysis Deck.

This ends the Preprocessing section.

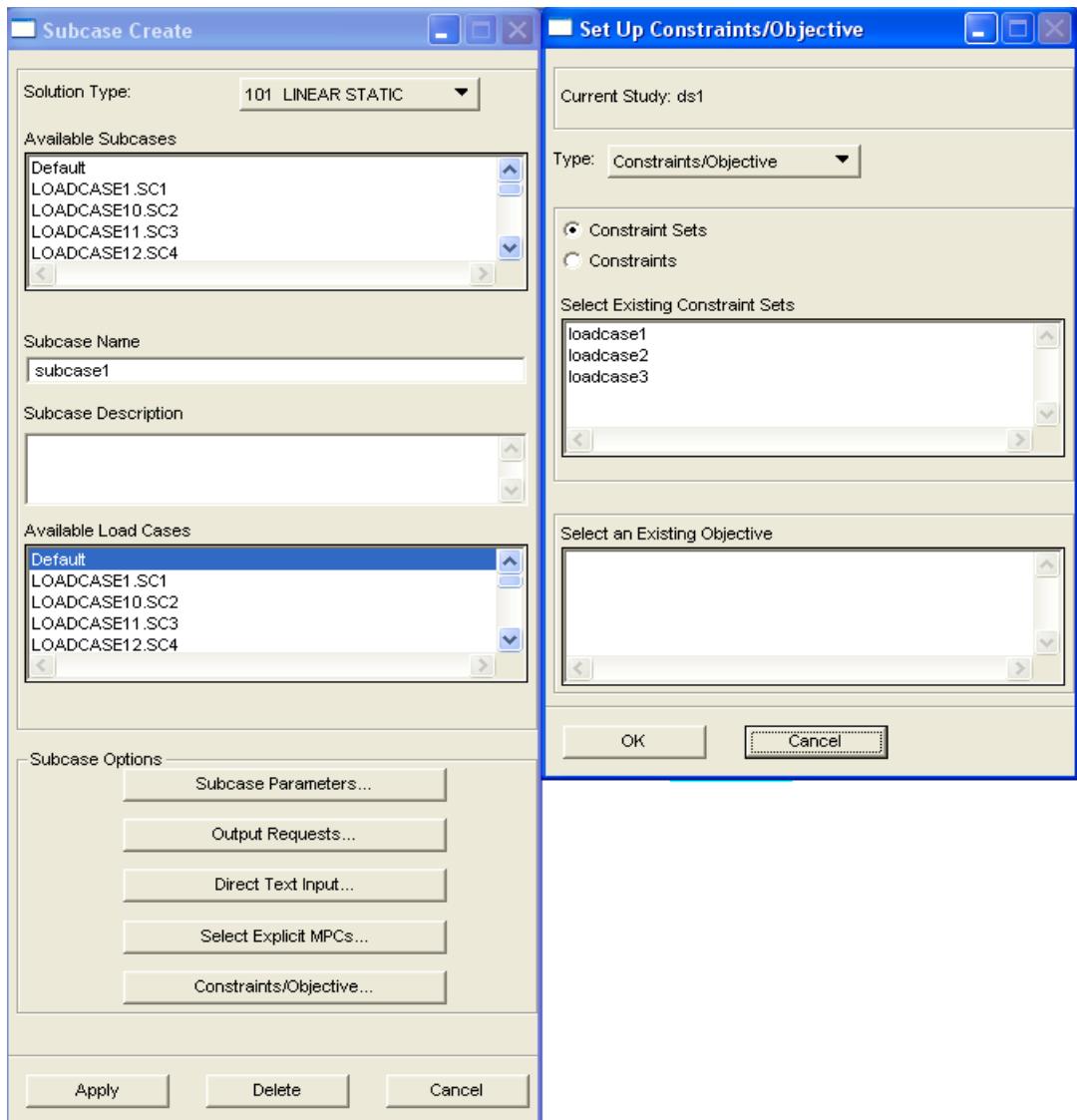


Figure 7-48 Creation of Subcases

Now run the deck with MSC Nastran which will generate the (.des) elemental density distribution file.

Creating Iso Surfaces using Patran Insight

This section illustrates how to use the Insight tool to create a smooth topology and iso surface(s) for density.

Click Tools, Design Study, Post Process and then select **Action**: Read Results. Then click the **Select Results File** tab to select the required .des file.

Step 1: Setting Insight Preferences for better Visual effect

- Click the **Insight** button on the toolbar. Click the **Preferences** menu and select **Insight**
- In the Insight Preferences form, select **Display Method**: Shaded; **Edge Color**: Black, **Face Color**: Gray; **Background Color**: White
- Click **Apply**.

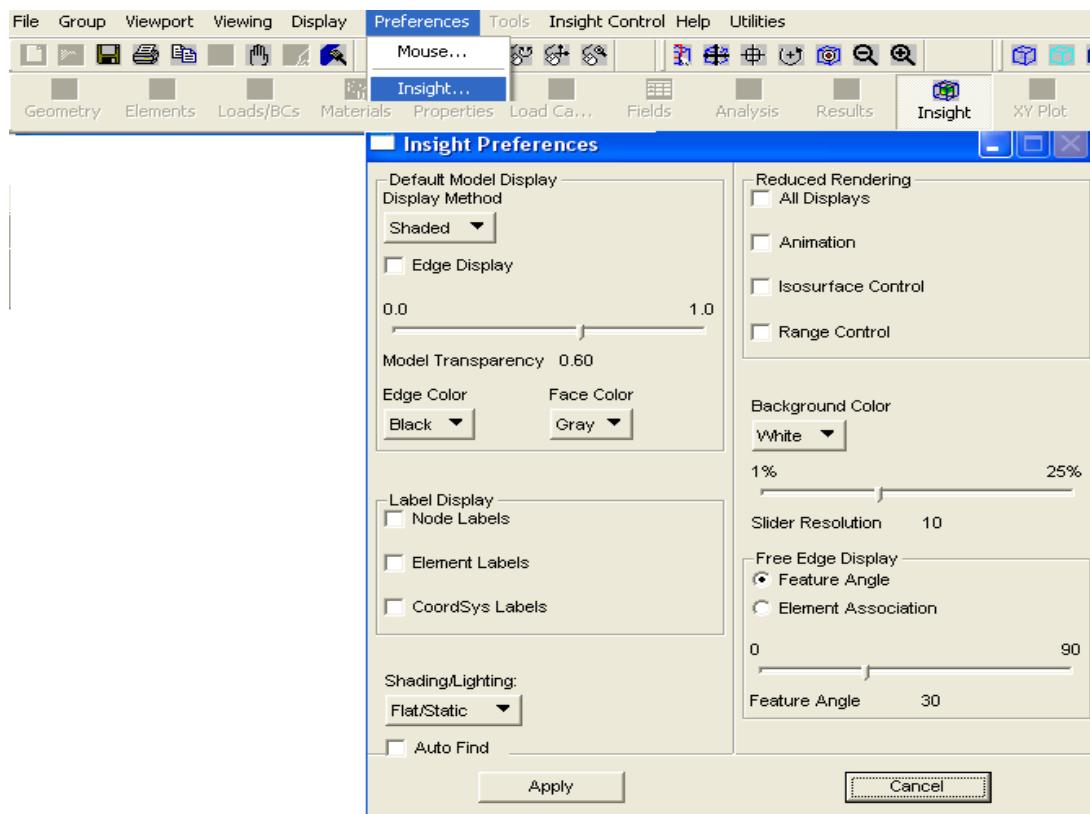


Figure 7-49 Insight Preferences

Step 2: Initiating the Insight Tools for creating Isosurface

- Click **Insight**.
- In the Insight Tools form, select **Action**: Create; **Tool**: Isosurface.



- c. Select the Result radio button in the Isosurface Value section.
- d. Click the **Results Selection** tab.
- e. In the Results Selection form, select one or more Current Load Case(s). Click the **Update Results** tab. Following this action, results will be updated into the Isosurface Result list.
- f. Select Topology Optimization, Element Density Distribution from the Isosurface Results list.
- g. Click the **Result Options** tab.
- h. Select the last load case from the Select Default Load Case list.
- i. Click the **Isovalue Setup** tab.
- j. Input an isovalue directly into the Isovalue box or through the slider bar.
- k. Click **OK** to close the Results Selection form.
- l. Click the **Isosurface Attributes** tab.
- m. In the form, select **Color:** Red, select the Clip in the isosurface box . Select **Display:** Free Edge and Shaded.
- n. Click **OK**.
- o. Click **Apply**.



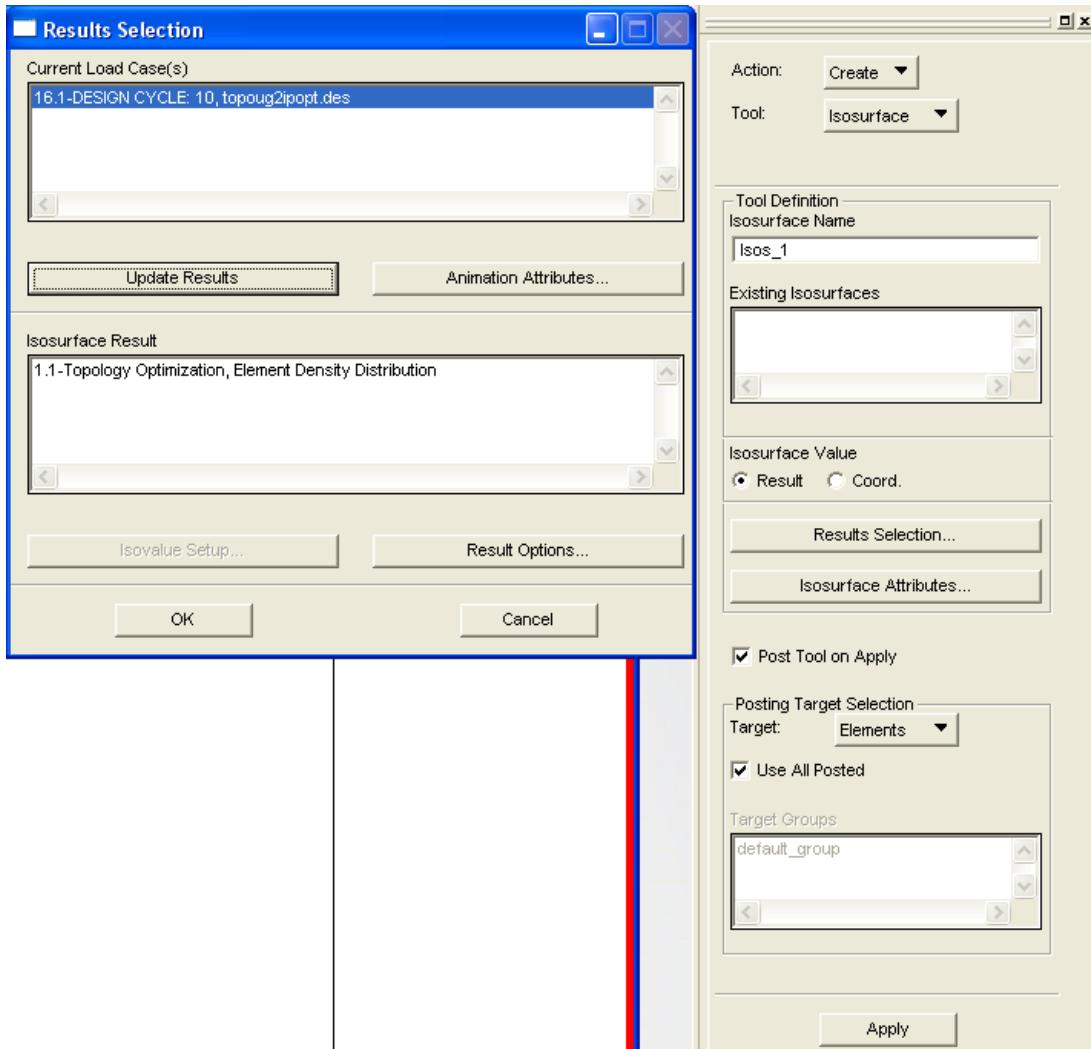


Figure 7-50 Creating Isosurface



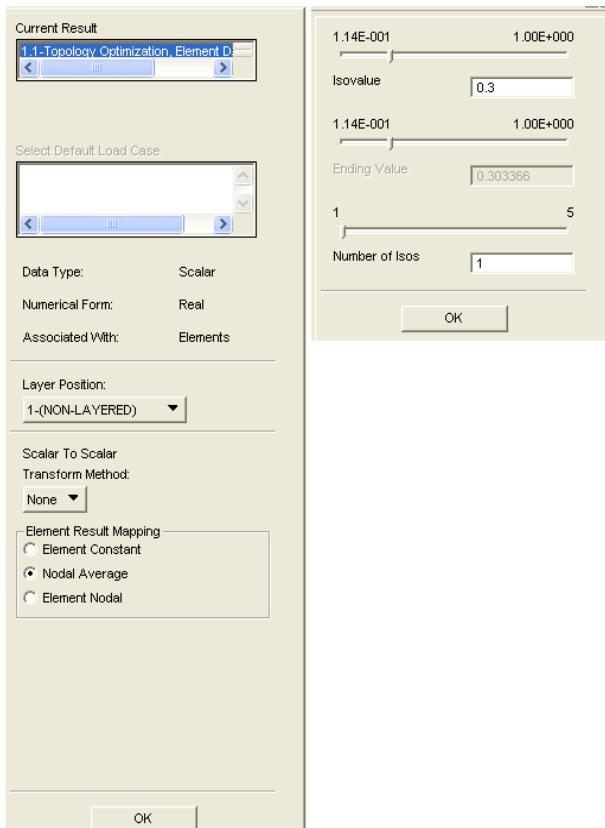


Figure 7-51 Setting Isovalue



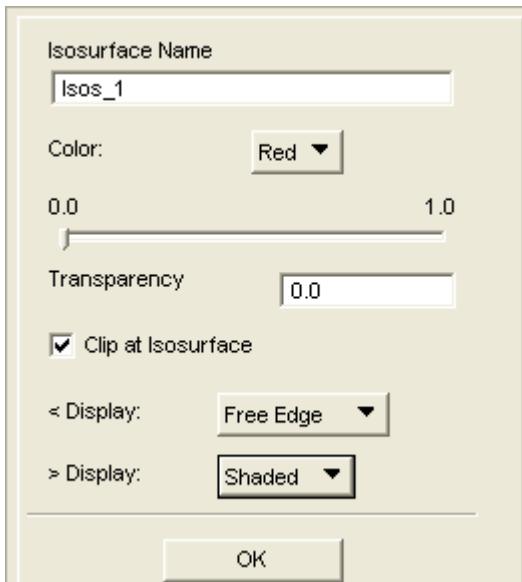


Figure 7-52 Isosurface Attributes

The following Isosurface appears on clicking Apply.

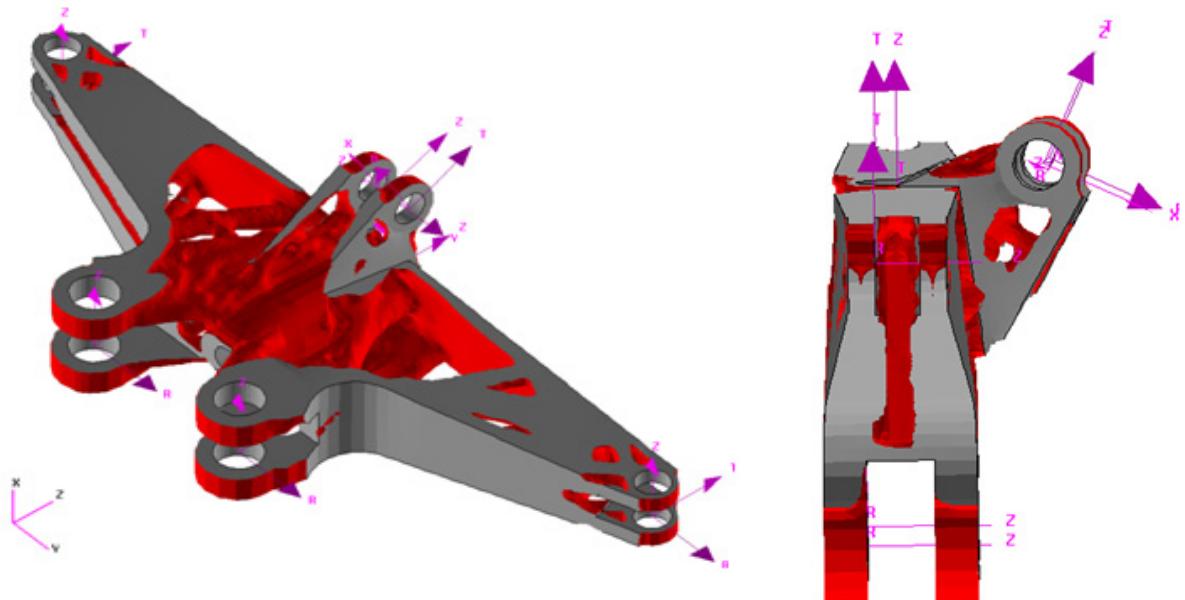


Figure 7-53 Isosurface



Step 3: Insight Control

This enables user to see different isosurfaces dynamically

- a. Click **Insight Control**→Isosurface Controls
- b. **Form Action:** Immediate
- c. Slide the Isovalue slider bar to dynamically view the isosurface changes.



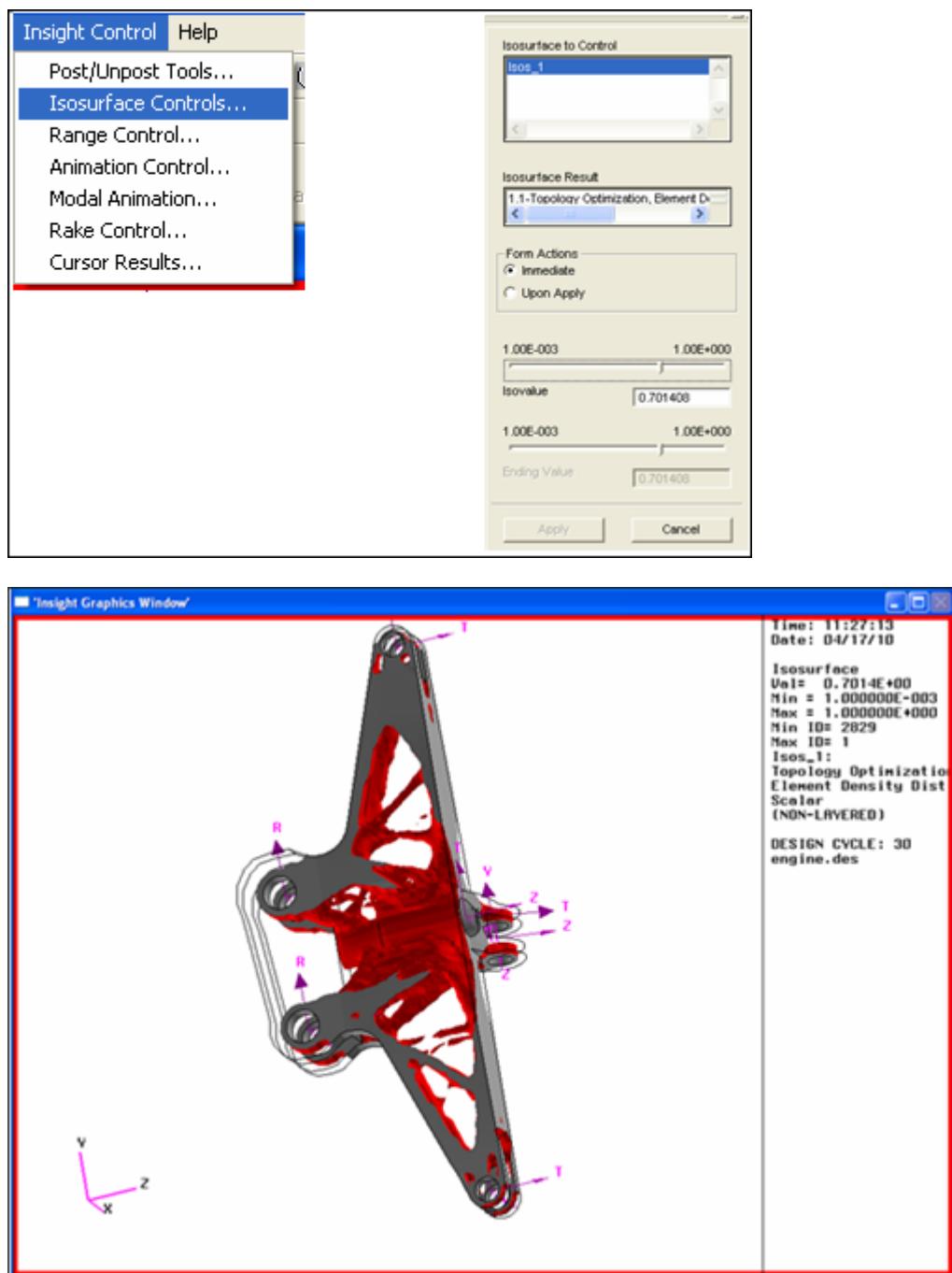


Figure 7-54 Isosurface Control

Topology Optimization Input Deck for MSC Nastran

The input data related to the Topology Optimization model is given in Listing 7-8.

The TOPVAR entries define five topological design parts with XINIT (initial design)=0.3 that matches the mass target so the initial design is feasible (reduce CPU time spent on optimizer).

In order for a structural response to be used either as an objective or a constraint, it first must be identified on a DRESP1 Bulk Data entry. The DRESP1 entries 200-213, for example, identify the compliance. DRSPAN and SET Case Control Commands are then used to select one compliance DRESP1 entry for each subcase that is used in DRESP2=1000 response. The equation response DRESP2=1000 with the attribute FUNC=AVG spans all subcases to calculate averaged compliance of the structure. A Case Control Command DESOBJ selects DRESP2=1000 to be an objective.

DRESP1=500 defines a fractional mass response. This mass target is imposed by the upper bound on the DCONSTR=50 entry. As always, fractional mass constraints should be applied at the global level in a design optimization by using DESGLB. Separate DRESP1 entries 1 -5 identify displacements responses at grid points. There responses are constrained by the bounds set using a corresponding set of DCONSTR entries.

Listing 7-8 Input File for Engine Mount

```

analysis=statics
set 1 = 200
set 2 = 201
set 3 = 202
set 4 = 203
set 5 = 204
set 6 = 205
set 7 = 206
set 8 = 207
set 9 = 208
set 10 = 209
set 11 = 210
set 12 = 211
set 13 = 212
set 14 = 213
DESOBJ = 1000
DESGLB = 50
DESSUB = 1
$ Direct Text Input for Global Case Control Data
$ =====
$ =====
SUBCASE 1
LOAD = 1
DRSPAN = 1
SUBCASE 2
LOAD = 2
DRSPAN = 2
SUBCASE 3
LOAD = 3
DRSPAN = 3
SUBCASE 4
LOAD = 4
DRSPAN = 4
SUBCASE 5
LOAD = 5
DRSPAN = 5
SUBCASE 6
LOAD = 6
DRSPAN = 6

```



```

SUBCASE 7
    LOAD = 7
DRSPAN = 7
SUBCASE 8
    LOAD = 8
DRSPAN = 8
SUBCASE 9
    LOAD = 9
DRSPAN = 9
SUBCASE 10
    LOAD = 10
DRSPAN = 10
SUBCASE 11
    LOAD = 11
DRSPAN = 11
SUBCASE 12
    LOAD = 12
DRSPAN = 12
SUBCASE 13
    LOAD = 13
DRSPAN = 13
SUBCASE 14
    LOAD = 14
DRSPAN = 14
$=====
BEGIN BULK
$
$23456781234567812345678123456781234567812345678123456781234567812345678
DCONSTR 1      1      -6.      6.0
DCONSTR 1      2      -6.      6.0
DCONSTR 1      3      -6.      6.0
DCONSTR 1      4      -6.      6.0
DCONSTR 1      5      -6.      6.0
DCONSTR 50     500      .3
TOPVAR, 1 , psolid, Psolid, .3, , , , 1
TOPVAR, 2 , psolid2, Psolid, .3, , , , 2
TOPVAR, 3 , psolid3, Psolid, .3, , , , 3
TOPVAR, 4 , psolid8, Psolid, .3, , , , 8
TOPVAR, 5 , psolid9, Psolid, .3, , , , 9
TOPVAR, 6 , psolid10, Psolid, .3, , , , 10
$23456781234567812345678123456781234567812345678123456781234567812345678
DRESP1 500     w      FRMASS
DRESP1 1       d      disp      123      76095
DRESP1 2       d1     disp      123      76096
DRESP1 3       d2     disp      123      76419
DRESP1 4       d3     disp      123      76420
DRESP1 5       d4     disp      123      76421
$23456781234567812345678123456781234567812345678123456781234567812345678
DRESP1, 200, COMP1, COMP
DRESP1, 201, COMP2, COMP
DRESP1, 202, COMP3, COMP
DRESP1, 203, COMP4, COMP
DRESP1, 204, COMP5, COMP
DRESP1, 205, COMP6, COMP
DRESP1, 206, COMP7, COMP
DRESP1, 207, COMP8, COMP
DRESP1, 208, COMP9, COMP
DRESP1, 209, COMP10, COMP
DRESP1, 210, COMP11, COMP
DRESP1, 211, COMP12, COMP
DRESP1, 212, COMP13, COMP
DRESP1, 213, COMP14, COMP
$23456781234567812345678123456781234567812345678123456781234567812345678
DRESP2 1000    COMPL AVG
    DRESP1 200      201      202      203      204      205      206
                    207      208      209      210      211      212      213

```



A topology result shown in [Figure 7-55](#) is obtained by MSC Nastran. The Topology Optimization design proposal is smoothed by Patran.

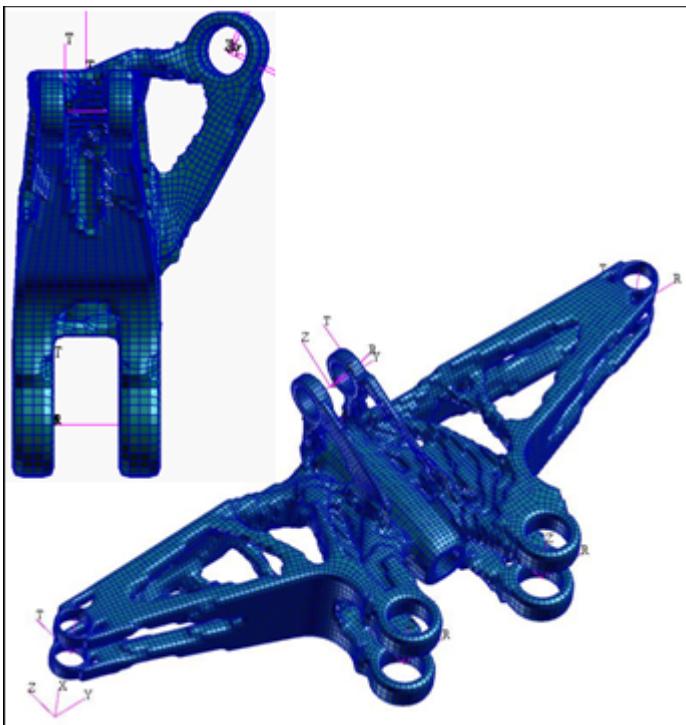
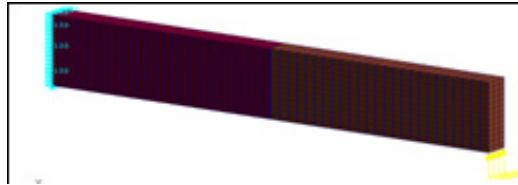


Figure 7-55 Front-Mount-Beam Topology Optimization Proposal



Topology Optimization with Glued Contact

Summary

ATTRIBUTE	VALUE
Title	Glued Contact
Topology Optimization features	Topology Optimization with glued contact
Geometry	
Material	$E = 2.1 \text{ E+11 Pa}$, $\mu=0.3$, and $\text{RHO}=7800$
Analysis	Statics
Boundary conditions	Fixed at one end
Applied loads	Distributed upward load at lower edge of free end
Element types	CHEXA
Topology result	Material distribution

Introduction

This problem has two solids glued together along a transverse plane to form a cantilever. The composite cantilever is used to demonstrate Topology Optimization with glued contact. The objective is to minimize the compliance subject to mass constraint of 0.3 (70% weight reduction). The loading and boundary conditions are shown in [Figure 7-56](#). The structure is modeled with 1683 CHEXA elements of PSOLID=1 property and 975 CHEXA elements with PSOLID=2 property. (Refer topoug5.dat in TPL).



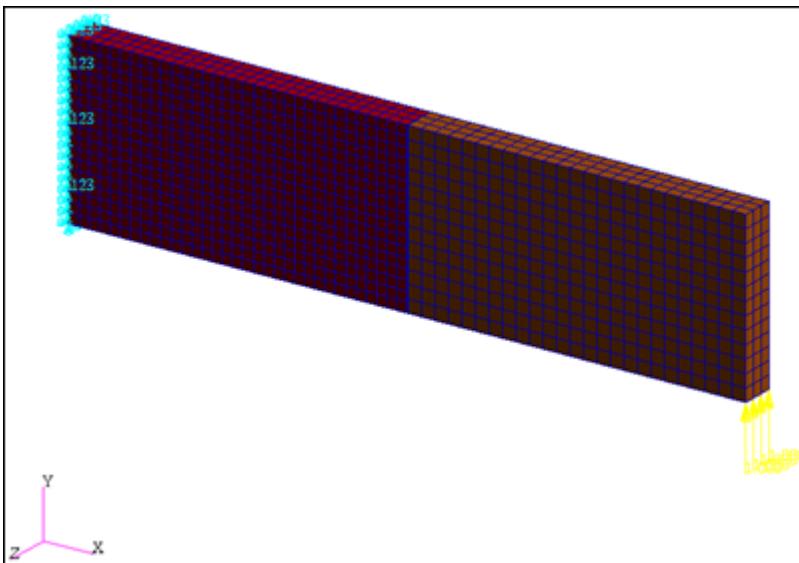


Figure 7-56 Composite Cantilever with two solids permanently glued

Solution Requirements

Design Model Description

Objective: Minimize compliance

Topology design region: PSOLID1 and PSOLID2

Constraints: Mass target = 0.3 (i.e., mass savings 70%)

The solution demonstrates:

- Topology Optimization with glued contact

Optimization Solution

The input data for this example related to Topology Optimization model is given in [Listing 7-10](#). Two TOPVAR entries are used to define two topological design regions identified by PSOLID=1 and PSOLID=2. XINIT=0.3 on the TOPVAR entries match the mass target constraint so that the initial design is feasible. The rest of the values on the TOPVAR entry are default values that are recommended for general Topology Optimization applications. Type one design responses DRESP1 = 2 and 10 identify fractional mass and compliance respectively. DCONSTR= 1 specifies the mass target. DESOBJ=10 in Case Control Command selects DRESP1=10 entry to be used as a design objective (minimization as default) and DESGLB selects the design constraint DCONSTR= 1 to be applied in this Topology Optimization task.

Case Control Command BCONTACT =888 selects the Bulk Data Entry BCTABLE. Value 1 in field 5 of first line in BCTABLE entry indicates that 1 set of slave/master entries is entered. "Slave" indicates



touching body and "master" indicates touched body. The presence of BCONTACT above the Subcase and value of 1 in field 8 (IGLUE) of "Slave" line in BCTABLE entry indicate that there is Permanent Glued Contact between the two bodies. The first entries 1001 and 2001 in "Slave" and "Master" lines respectively in BCTABLE entry are referenced by the two BCBODY entries with the corresponding IDs. Field 5 in BCBODY entries contains the IDs of BSURF entries which define the deformable surfaces identified by element IDs. In this problem deformations are small and linear.

Listing 7-9 Input File for Glued Contact

```

DESOBJ = 10
DESGLB = 1
BCONTACT = 888
smethod=element
ANALYSIS = STATICS
$ Direct Text Input for Global Case Control Data
SUBCASE 1
$ Subcase name : Default
SUBTITLE=Default
SPC = 2
LOAD = 2
DRSPAN = 1
BEGIN BULK
$-----2-----3-----4-----5-----6-----7-----8-----9-----0
BCTABLE 888
          1
+
+      SLAVE   1001    0.1
+
+      MASTER   2001
+
$ .....
$
DCONSTR 1      2      .3
TOPVAR   1      PSOLID  PSOLID .3
TOPVAR   2      PSOLID  PSOLID .3
DRESP1   2      FRM     FRCMASS
DRESP1   10     COMP    COMP
$ Direct Text Input for Bulk Data
$ Elements and Element Properties for region : p1
PSOLID  1      1      0
BCBODY   1001   3D     DEFORM  3      0
BSURF    3      115    116    117    118    119    120    121
          122    123    124    125    126    127    128    129
+
$ .....
BCBODY   2001   3D     DEFORM  4      0
BSURF    4      1798   1799   1800   1801   1802   1803   1804
          1805   1806   1807   1808   1809   1810   1811   1812
+
$ .....

```



Figure 7-57 shows the topology optimized result by using Patran.

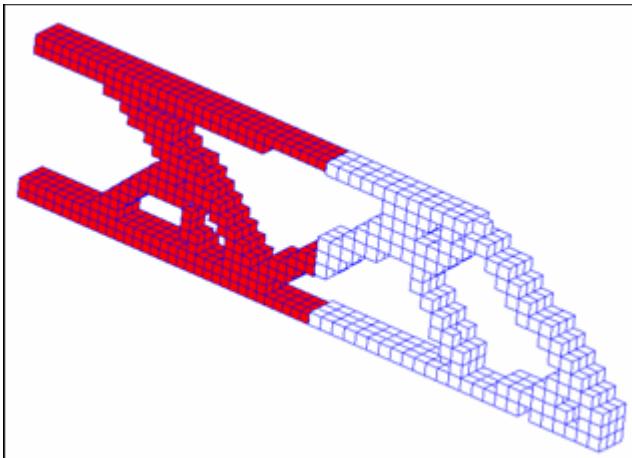


Figure 7-57 Glued Contact Topology Design

We can also set different fractional mass constraints for the two topological design regions. We will do this using Patran.

Step 1: Import the deck (model including contact information)

Step 2: Set up deck using Toptomize

- a. Click **Analysis**. Select **Create**: Toptomize, **Object**: Entire Model, **Method**: Analysis Deck.
- b. Click the **Objectives and Constraints** tab.
- c. In the Objectives and Constraints form, select **Type**: Topology, **Objective Function**: Minimize Compliance and **Constraint Target**: Mass Fraction.



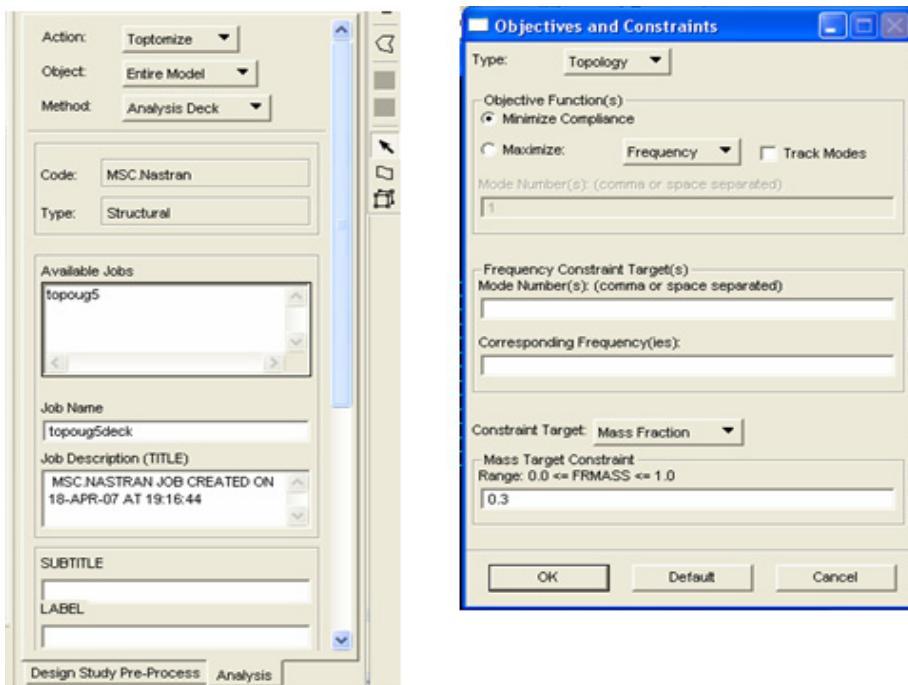


Figure 7-58 Initiating Toptomize

- d. Click the **Design Domain** tab.
- e. In the Design Domain form, click p1 and p2 from Valid Properties list .
- f. Change the fractional mass target of p1 to 0.2 and of p2 to 0.4 on the Design Domain Selected Properties table.
- g. Click **OK**.
- h. Click the **Subcase Select** tab and select the default subcase. Click **OK**.
- i. Click **Apply**.



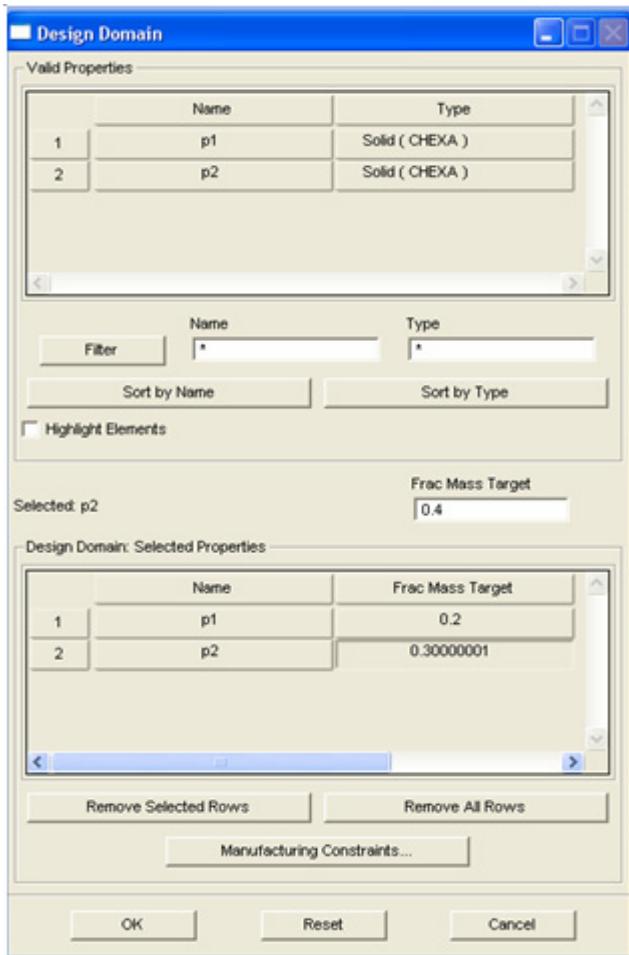


Figure 7-59 Setting multiple mass targets



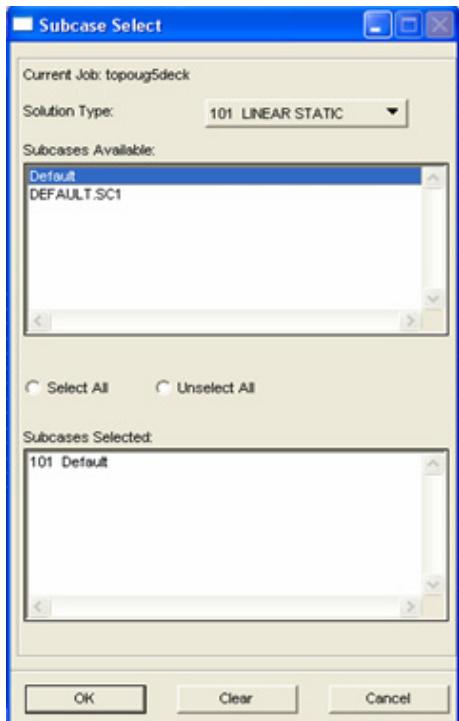


Figure 7-60 Selecting Subcase



Topometry Optimization

Introduction

Topometry optimization is an element-by-element sizing optimization. Unlike conventional sizing optimization where all elements referencing a property entry are grouped as one design variable, each designable element has an independent design variable in topometry optimization. Since element-by-element optimization has many design variables, it may find a better design than conventional sizing optimization. In typical sizing optimization, you can use the design variable Bulk Data entry DESVAR and the relation of model property and design variables Bulk Data entry DVxREL1 to support element-by-element sizing optimization. However, with this approach you must generate a unique property data entry for each element and perhaps prepare thousands of DESVAR and DVPREL1 entries. With the topometry optimization capability, you can utilize Bulk Data entry, TOMVAR, to select designable regions (model property or material property identification number), design parameters (such as thickness of PSHELLs, or Young's Modulus of materials), input initial values, lower and upper bounds to perform element-by-element sizing optimization. The program internally generates DESVAR and DVPREL1 (and/or DVMREL1) for each designed element. The implementation provides a very simple user interface to do element-by-element sizing design optimization. In addition, topometry optimization supports the fully stressed design algorithm in MSC Nastran. FSD is very efficient for certain problems with many stress constraints.

Topometry optimization can be applied to all elements that can be resized through Bulk Data entries DVPREL1 and DVMREL1. Those element types include not only volume-based elements like CQUAD4 but also non-volume elements like CWELD, CBUSH, and CFAST.

Topology optimization is another element-by-element optimization technology. However, topology optimization and topometry optimization are fundamentally different. Topology optimization is a “0” or “1” discrete element-by-element optimization methodology. Topology optimization can be used to decide which element should be retained and which element should be discarded from the design space. On the other hand, topometry optimization aims to get a continuous variation of the designed properties. Although topometry optimization is not recommended for topology optimization tasks, it is observed topometry optimization can be used to get “similar topological results” for some cases. It is particularly useful for non-structural elements like CELAS, CFAST, and CBUSH that MSC Nastran topology optimization does not support.

In a single optimization problem, you can resize (or shape, topology) certain properties while topometry optimizing other properties.

Benefits

- Topometry optimization is easy-to-use. One TOMVAR Bulk Data entry replaces many thousands of DESVAR and DVxREL1 entries for large element-by-element design optimization problems.
- Topometry optimization is good to identify critical design regions.



- Topometry optimization is good to locate where to add/or remove material to improve structural performance.
- Topometry optimization is good for finding the optimal location of spot welds. In particular, topometry optimization is very useful for some properties that MSC Nastran topology does not support; for example, PDAMP, PELAS, PMASS, PBUSH, PVISC, PGAP, PACBAR, and PFAST.

TOMVAR

The TOMVAR Bulk Data entry is used to select a topometry designable region and designed property name. The initial, lower, and upper bound of the designed property value are also specified on the topometry entry. The program automatically generates one design variable DV_i for each element referencing a property PID. The relationship between design variables DV_i and the element property P_i given by

$$P_i = DV_i \quad i = 1, NE$$

$$XLB \leq DV_i \leq XUB$$

where P_i is the analysis model property value for the i th element. NE is the total number of elements referenced by the property PID. The user must input an initial value (such as the analysis model input property value). The default of lower bound (XLB) on DV_i is $0.5 \cdot DV_i$, and default of upper bound on DV_i (XUB) is $1.5 \cdot DV_i$.

The topometry Bulk Data entry is:

Format

1	2	3	4	5	6	7	8	9	10
TOMVAR	ID	TYPE	PID	PNAME /FID	XINIT	XLB	XUB	DELXV	
	DLINK	TID	C0	C1					
	DDVAL	DSVID							

Example 1

Design all element's thickness referencing PSHELL ID = 5 with initial design = 10.0 ($t_0 = 10.0$ input element thickness), lower bound $0.5 \cdot t_0$ and upper bound $1.5 \cdot t_0$.

TOMVAR	10	PSHELL	5	T	10.0				
--------	----	--------	---	---	------	--	--	--	--

Example 2

Design all element's Young Modulus referred by PSHELL ID = 100 with initial design XINIT = 3.E+5, XLB=1.0, and XUB= 1.0E+6.



TOMVAR	10	PSHELL	100	E	3.E+5	1.0	1.E+6		
--------	----	--------	-----	---	-------	-----	-------	--	--

Field	Contents
ID	Unique topometry design region identification number. (Integer > 0)
TYPE	Property entry type. Used with PID to identify the elements to be designed. See Remark 2. (Character: “PBAR”, “PSHELL”, ‘PSOLID’, and “PCOMP”, etc.)
PID	Property entry identifier (Integer > 0). This PID must be unique for PIDs referenced by other TOPVAR, DVPREL1, DVPREL2, DVMREL1, and DVMREL2 entries. Topometry, topology, and sizing variables cannot share the same properties. (Integer > 0). Combined topometry, topology, topography, sizing, and shape variables are allowed.
PNAME/FID	Property name or property material name, such as “T”, “A”, “E”, and “GE”, or field position of the property entry or word position in the element property table of the analysis model. Property names that begin with an integer such as 12I/T**3 may only be referenced by field position. See Remark 2. (Character or Integer > 0.)
XINIT	Initial value. (Real or blank, no Default). Typically, XINIT is defined to match the mass target constraint (so the initial design does not have violated constraints) or the analysis model input property value.
XLB	Lower bound. (Real or blank; Default = blank) . The default is XLB=0.5*XINIT.
XUB	Upper bound . (Real or blank; Default = blank). The default is XLB=1.5*XINIT.
DELXV	Fractional change allowed for the design variable during approximate optimization. See Remark 3. (Real > 0.0; Default = 0.5)
DDVAL	Indicates that this line defines discrete TOMVAR variables.
DSVID	DDVAL entry identifier. (Integer > 0)
DLINK	Indicates that this line relates a ply thickness to another ply thickness.
TID	TOMVAR entry identifier. (Integer > 0)
C0	Constant term. (Real; Default = 0.0)
C1	Coefficient term. (Real; no Default)

Remarks:

1. Multiple TOMVAR's are allowed in a single file.
2. Property name or FID > 0 can be used for element property values just like a Bulk Data entry DVPREL1. Only property name can be used for material property values like DVMREL1. If a property name is shared by both property and material (such as “A” for PROD and MAT1), this name is taken as a material name. The user must provide a FID for property name (FID=4 for PROD). PCOMPG, PBEAML, PBARL, PBMSECT, PBRSECT are not supported. If material property name is selected, PSHELL (with multiple MID inputs) must reference a single material ID.



3. Combined topometry, topography, topology, sizing, and shape optimization is supported in a single file. However, topometry and topology cannot reference the same property ID. It is possible to topometry certain elements while sizing others. It is allowed, but rare, to simultaneously design the same elements with topometry and desvar (sizing and/or shape) variables but topometry and sizing cannot reference the same property name.
4. Topometry optimization with element response constraints are slow due to many design variables. In this case, fully stressed design (FSD) can be used for certain problems
5. Parameters DESPCH and DESPCH1 specify when the topometry optimized design values are written to the element result history file jobname.des that can be imported to Patran and other post-processor to view topometry optimized results.

Output

A regular SOL 200 summary table is produced. In addition, a Patran element result file jobname.des contains the optimal design values for each element. This Patran element result file can be imported to Patran or other post-processors to display topometry optimization results.

Guidelines and Limitations

- MSCADS, method=4 (SUMT) is recommended for topology, topometry and topography optimization with many constraints through the optimization control Bulk Data entry DOPTPRM.
- Since SOL 200 adjoint design sensitivity analysis method does not support element responses (such as stress), a direct design sensitivity analysis method is automatically selected for problems with element response constraints. In this case, topometry optimization with element response constraints are slow due to many design variables. Fully stressed design (FSD) can be used for certain problems.
- Topometry optimization can be used for analysis model properties PDAMP, PELAS, PMASS, PBUSH, PVISC, PGAP, NSM, NSM1, PACBAR and PFAST. Topology optimization is limited to analysis properties that can reference material property MAT1.
- $P_2 \geq 13$ on DOPTPRM prints design variables in *.f06.

Example 1 - Three-bar Truss (tomex1.dat)

A simple sizing optimization example three-bar truss (a TPL file DSOUG1.dat) is adapted here to demonstrate topometry optimization solved by the fully stressed design algorithm. [Figure 7-61](#) shows the three-bar truss that must withstand two separate loading conditions. The objective is to minimize structural weight subject to displacement and stress constraints. The sizing design variables are the cross-sectional areas. The detailed descriptions of analysis model and design optimization model can be seen in the three bar truss [Figure 7-61](#).



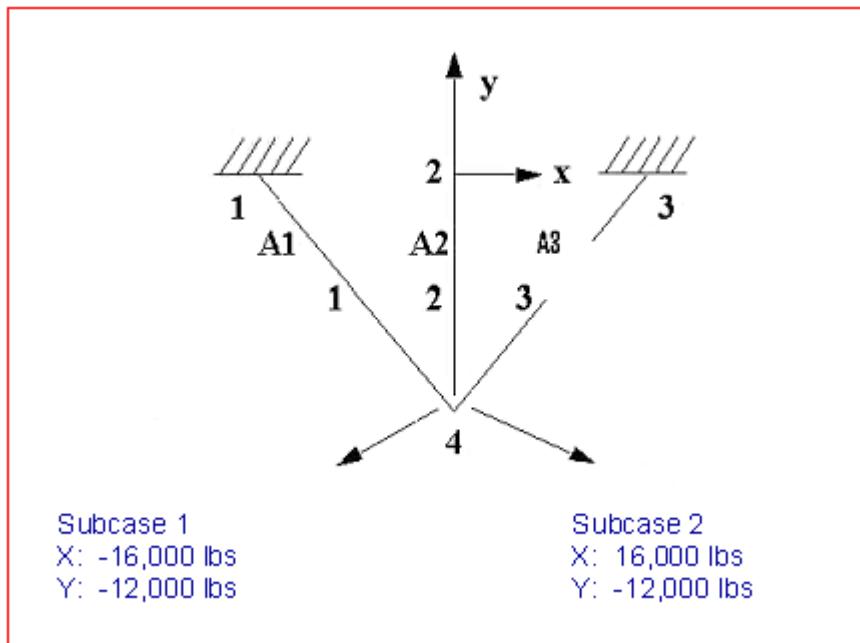


Figure 7-61 Three Bar Truss

The goal of this example is to show an alternate method of setting design variables by a TOMVAR entry. The objective and constraints are not changed. In conventional sizing optimization, the set of DESVAR and DVPREL1 entries define the relations $A_i = 1.0X_i$ ($i=1, 2, 3$) where A is the rod element cross-sectional area and X is the design variable. In DSOUG1.dat, we have:

```
$...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
$
$...DEFINITION OF DESIGN VARIABLE TO ANALYSIS MODEL PARAMETER $RELATIONS
$DVPREL1 ID      TYPE     PID      NAME     PMIN     PMAX     C0      +
#+      DVID1    COEF1    DVID2    COEF2    ...
DVPREL1 10      PROD     11      A
1          1.0
DVPREL1 20      PROD     12      A
2          1.0
DVPREL1 30      PROD     13      A
3          1.0
```

In DSOUG1.dat, rod elements 11 and 12 have different property groups. Then, the DLINK entry is used to explicitly link the design variables 1 and 3 together. In this example, we try to do element-by-element optimization. Thus, we take three design variables (rod element cross-sectional areas) as independent variables. The rod elements 1 and 3 have the same property group (PROD=1). TOMVAR entry 1 (Listing 7-10) is used to define two independent design variables with an initial value = 1.0 (and element cross-sectional area = 1.0) for rod element 11 and 13 respectively. This is equivalent to four entries in DSOUG1.dat:



DESVAR	1	A1	1.0	0.1	100.0
DESVAR	3	A3	2.0	0.1	100.0
DVPREL1	10	PROD	11	A	
	1		1.0		
DVPREL1	30	PROD	13	A	
	3		1.0		

TOMVAR entry 2 ([Listing 7-10](#)) is used to define one independent design variable with an initial value = 2.0 (and element cross-sectional area = 2.0) for rod element 12. This is equivalent to two entries in DSOUG1.dat:

DESVAR	2	A2	2.0	0.1	100.0
DVPREL1	20	PROD	12	A	
	2	1.0			

Input

The input data for this example is given in Listing 7-10.

Listing 7-10 Input File for Example 1

```

ID MSC TOMEX1 $
TIME 10 $
SOL 200      $ OPTIMIZATION
CEND
TITLE = THREE BAR TRUSS TOPOMETRY OPTIMIZATION
SUBTITLE = 3 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
SPC1  100    123456  1      THRU    3
$ ELEMENT DATA
CROD  1      11      1      4
CROD  2      12      2      4
CROD  3      11      3      4
$ PROPERTY DATA
PROD  11      1      1.0
PROD  12      1      2.0
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 TOPOMETRY DESIGN DEFINITION
$TOMVAR, ID, PRYPE, PID, PNAME, XINIT, XLB, XUB, DELXV(OPTIONAL)
TOMVAR, 1 , PROD, 11, 4      , 1., .1 , 100.0
TOMVAR, 2 , PROD, 12, 4      , 2., .1 , 100.0
$...
$...STRUCTURAL RESPONSE IDENTIFICATION
$DRESP1 ID      LABEL   RTYPE   PTYPE   REGION ATT AATTB ATT1 +
$+    ATT2    ...
DRESP1 20      W       WEIGHT
DRESP1 21      U4      DISP
DRESP1 23      S1      STRESS  PROD
                12
                2
                11
$...
$...CONSTRAINTS
$DCONSTR DCID  RID      LALLOW  UALLOW
DCONSTR 21    21      -0.20   0.20
DCONSTR 21    23      -15000. 20000.
$...
$...OPTIMIZATION CONTROL (FULLY STRESSED DESIGN):
$-
DOPTPRM FSDMAX 20      DESMAX 0      P1      1      P2      15
$-
$.....2.....3.....4.....5.....6.....7.....8.....9.....0
ENDDATA

```

Output

A regular SOL 200 output can be found as:

```

*****
***** SUMMARY OF DESIGN CYCLE HISTORY *****
*****                                     *****
(HARD CONVERGENCE ACHIEVED)

NUMBER OF FINITE ELEMENT ANALYSES COMPLETED           17
NUMBER OF FULLY STRESSED DESIGN CYCLES COMPLETED      16
NUMBER OF OPTIMIZATIONS W.R.T. APPROXIMATE MODELS      0

----- OBJECTIVE AND MAXIMUM CONSTRAINT HISTORY -----
CYCLE NUMBER          OBJECTIVE FROM APPROXIMATE OPTIMIZATION          OBJECTIVE FROM EXACT ANALYSIS          FRACTIONAL ERROR OF APPROXIMATION          MAXIMUM VALUE OF CONSTRAINT
----- INITIAL          4.828427E+00
1              FSD          3.862742E+00                         N/A
2              FSD          3.225798E+00                         N/A
.....
.....
.....
16              FSD          2.741757E+00                         N/A
                               1.664062E-04

----- DESIGN VARIABLE HISTORY -----
INTERNAL| EXTERNAL | DV. ID | ELEMENT ID | LABEL | INITIAL : 1      : 2      : 3      : 4      : 5
----- 1   | 1   | TOMVAR | 1.0000E+00 : 8.0000E-01 : 6.8794E-01 : 6.8306E-01 : 6.9978E-01 : 7.2284E-01
2   | 3   | TOMVAR | 1.0000E+00 : 8.0000E-01 : 6.8794E-01 : 6.8306E-01 : 6.9978E-01 : 7.2284E-01
3   | 2   | TOMVAR | 2.0000E+00 : 1.6000E+00 : 1.2800E+00 : 1.0240E+00 : 8.1920E-01 : 6.5536E-01
-----
```



Example 2 – Car Model Topometry Design

A real complex example car body is used here to demonstrate topometry optimization for graphical post-processing. This example also shows that SOL 200 is able to deal with very large optimization problems. The objective is to minimize structural compliance and keep weight unchanged. SOL 200 produces an element thickness distribution file *.des that can be used by Patran or other post-processors to view topometry optimization results.

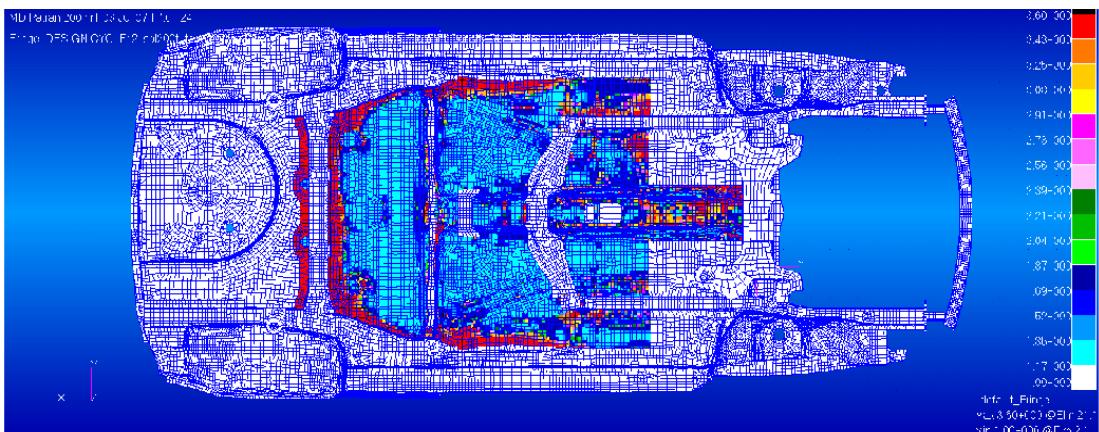
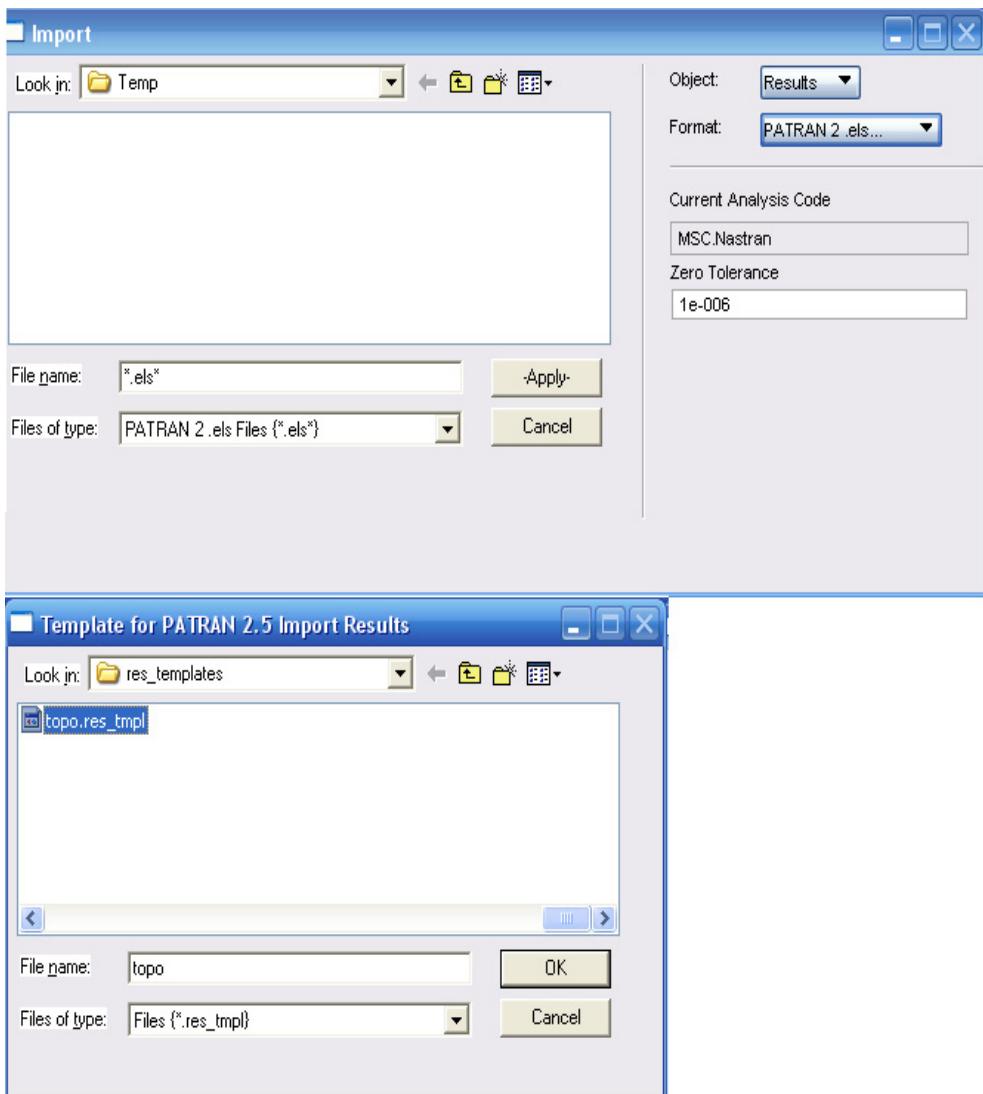


Figure 7-62 Optimal Thickness Distribution of Car Model

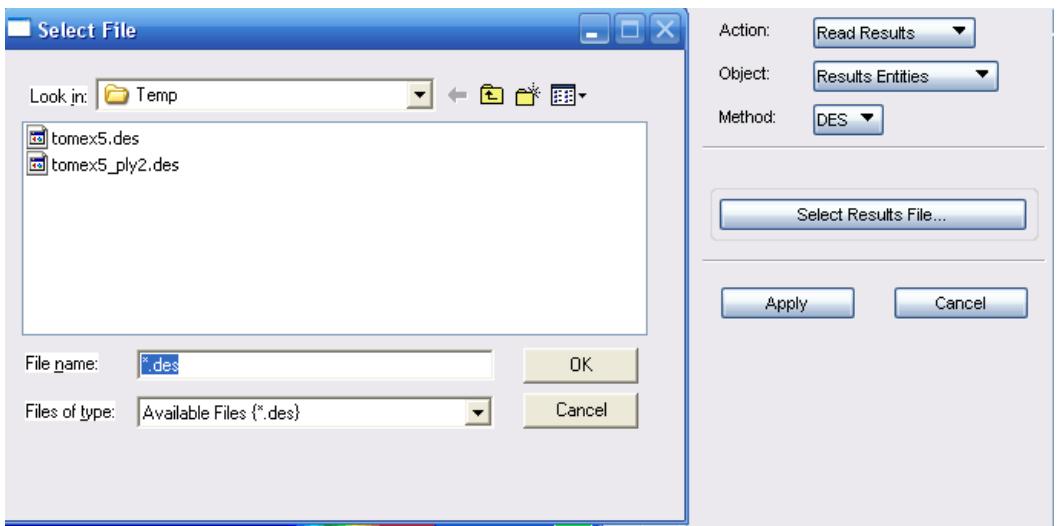
Post-processing Topometry Results

1. Read in the model through File->Import
2. Click File-> Import and in the Import form select Objects: Results and Format: Patran 2 .els...
3. In the Template for PATRAN 2.5 Import Results form select res_templates in Look in and then topo.res_tmpl from the list. Then click OK.
4. Click cancel to close the Import form





5. Read jobname.des (in case of plies, the results are jobname.ply000n): Click Tools->Design Study->Post Process and select Action: Read Results, Object: Result Entities, Method: DES and then click Select Result files. After selecting the desired .des or .ply000n file, click OK and then Apply.



6. Click Results and in the Results form select Action:Create, Object:Fringe, the desired design cycle in Select Result Cases, Topology Optimization, Element Density in Select Fringe Result and then click Apply. See image below.





Topography (Bead or Stamp) Optimization

Introduction

Topography optimization (also called bead or stamp optimization) is used to generate a design proposal for reinforcement bead patterns. Topography optimization is treated as a special shape optimization and built on SOL 200 shape optimization technology. In topography optimization, finite element grids are moved as normal vectors to the shell surface or the user's given direction. Internally, shape design variables and shape basis vectors are generated automatically based on your provided bead dimensions (minimum bead width, maximum bead height, and draw angle). Since many design variables are generated in the topography optimization, the adjoint design sensitivity analysis method and large scale optimizer play key roles in solving topography optimization problems.

Benefits

- Topography optimization is particularly powerful for designing sheet metal parts.
- Topography optimization can be used for all SOL 200 analysis types such as statics, normal modes, buckling, complex eigenvalue, dynamic frequency response, transient response and aeroelastic analyses.

BEADVAR

The BEADVAR Bulk Data entry is used to define topography design regions.

1	2	3	4	5	6	7	8	9	10
BEADVAR	ID	PTYPE	PID	MW	MH	ANG	BF	SKIP	
	“DESVAR”	NORM/XD	YD	ZD	CID	XLB	XUB	DELXV	
	“GRID”	NGSET	DGSET						

Field	Contents
ID	Unique topography design region identification number. (Integer > 0)
PTYPE	Property entry type. Used with PID to identify the element nodes to be designed. (Character: “PSHELL”, “PSHEAR”, “PCOMP”, or “PCOMPG”).
PID	Property entry identifier. See Remark 1. (Integer > 0)
MW	Minimum bead width. This parameter controls the width of the beads. The recommended value is between 1.5 and 2.5 times the average element width. See Remark 2. (Real > 0.0)
MH	Maximum bead height (Real > 0.0). This parameter sets the maximum height of the beads when XUB=1.0 (as Default). See Remark 2.

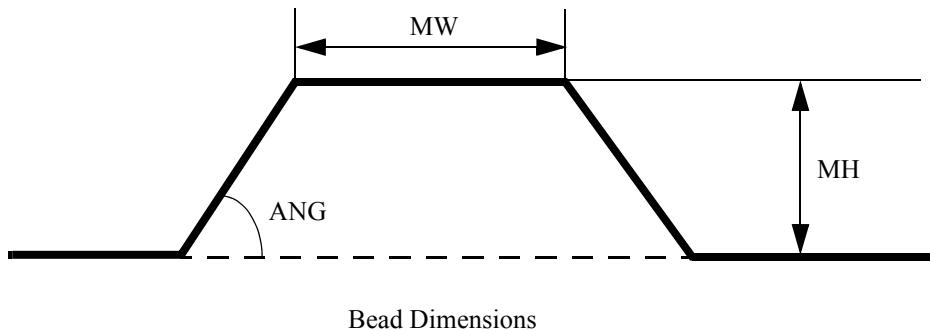


Field	Contents
ANG	Draw angle in degrees ($0.0 < \text{Real} < 90.0$). This parameter controls the angle of the sides of the beads. The recommended value is between 60 and 75 degrees.
BF	Buffer zone ('yes' or 'no'; Default='yes'). This parameter creates a buffer zone between elements in the topography design region and elements outside the design region when BF='yes'. See Remark 3.
SKIP	Boundary skip ("bc", "load", "both", or "none"; Default = "both"). This parameter indicates which element nodes are excluded from the design region. "bc" indicates all nodes referenced by "SPC" and "SPC1" are omitted from the design region. "load" indicates all nodes referenced by "FORCE", "FORCE1", "FORCE2", "MOMENT", "MOMENT1", "MOMENT2", and "SPCD" are omitted from the design region. "both" indicates nodes with either "bc" or "load" are omitted from the design region. "none" indicates all nodes associated with elements referencing PID specified in field 4 are in the design region.
"DESVAR"	Indicates that this line defines bead design variables that are automatically generated.
NORM/XD, YD, ZD	Bead vector (draw direction). Norm indicates the shape variables are created in the normal directions to the elements. If XD, YD, and ZD are provided, the shape variables are created in the direction specified by the xyz vector defied by XD/YD/ZD that is given in the basic coordinate system or CID. See Remark 4. (Character or Real, Default = blank = norm).
CID	Coordinate system ID used for specifying draw direction (Blank or Integer > 0 ; Default = blank = basic coordinate system)
XLB	Lower bound. ($\text{Real} < \text{XUB}$ or blank; Default = blank = 0.0). This ensures the lower bound on grid movement equal to XLB*MH. See Remark 5.
XUB	Upper bound. ($\text{Real} > \text{XLB}$ or blank; Default = 1.0). This sets the upper bound of the beads equal to XUB*MH. See Remark 5.
DELXV	Fractional change allowed for the design variable during approximate optimization. See Remark 3. (Real > 0.0 ; Default = 0.2)
"GRID"	Indicates this line defines what element nodes can be added and/or removed from topography design regions.
NGSET	All grids listed on Bulk Data entry SET1 = NGSET are removed from topography design regions.
DGSET	All grids listed on Bulk Data entry SET1 = DGSET are added to topography design regions.

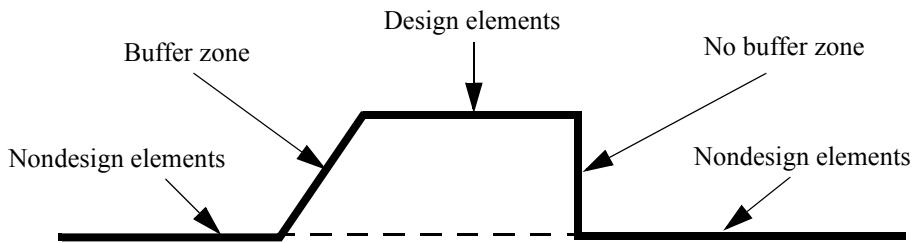


Remarks:

1. Multiple BEADVAR's are allowed in a single file. Combined topometry, topology, topography, sizing, and shape optimization is supported in a single file.
2. The user can provide allowable bead dimensions.

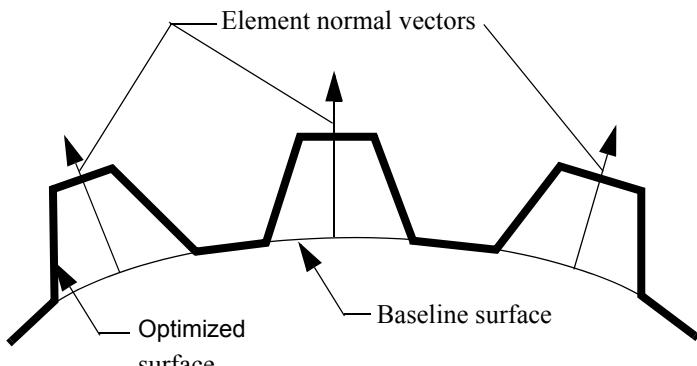


3. It is recommended to set buffer zone = yes to maintain a good quality of mesh during topography optimization.

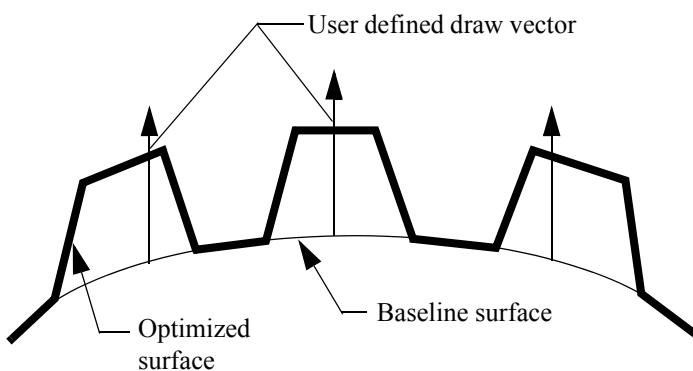


4. The grids moves in the normal direction. All element grids referenced by one BEADVAR entry must follow the right hand rule.





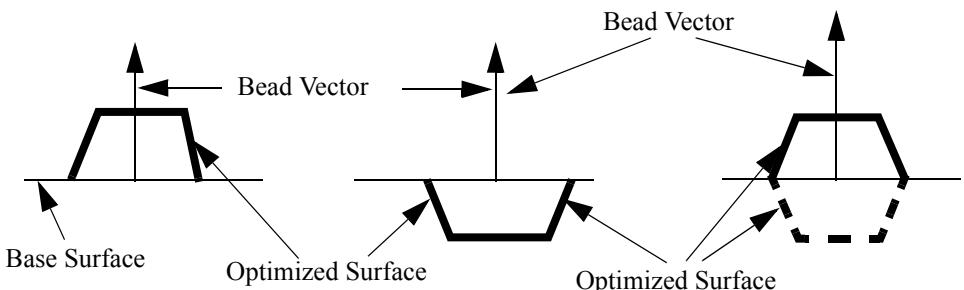
Element Normal



User's Provided Draw Direction

5. To force the grids to move only in the positive bead vector direction (one side of the surface), use $XLB = 0.0$. To force the grids to move only in the negative bead vector direction (another side of the surface), use $XUB = 0.0$. To allow grids to move in both positive and negative bead vector directions, use $XLB < 0.0$ and $XUB > 0.0$. For example,



(a) $\text{XLB} = 0.0$ and $\text{XUB} = 1.0$ (b) $\text{XLB} = -1.0$ and $\text{XUB} = 0.0$ (c) $\text{XLB} = -1.0$ and $\text{XUB} = 1.0$

6. The jobname.op2 has topography results (shape change) that can be viewed in Patran. The text file jobname.pch also has updated grid coordinates that can be copied to replace the grids in the original file, and imported to Patran on other post-processors to view topography optimization results.

Outputs

A regular SOL 200 design history summary table is produced. The jobname.op2 (with PARAM,POST,-1) and jobname.pch can be imported to Patran and other post-processors to view topography optimization results.

Guidelines and Limitations

- MSCADS, method=4 (SUMT) is recommended for many constraint problems through the optimization control Bulk Data entry DOPTPRM.
- Since SOL 200 adjoint design sensitivity analysis method does not support element responses (such as stress), a direct design sensitivity analysis method is automatically selected for problems with element response constraints. In this case, topography optimization with element response constraints is slow.
- Since adjoint design sensitivity analysis does not support rigid body elements (RBE1, RBE2, RBE3, RROD, RBAR, RTRPLT, RSPLINE), all grids connected to rigid body elements must be fixed in topography optimization for static and dynamic frequency response analyses.
- The minimum bead width and maximum bead height have significant effects on optimal designs. A smaller minimum bead width results in more small beads.
- Mesh distortion is a challenge for topography optimization. It is recommended that a relatively coarse mesh be used for highly curved areas.



- P2 ≥ 13 on DOPTPRM prints design variables in *.f06

Example 3 – A Square (togex1.dat)

A square model shown in Figure 7-63 is used to demonstrate topography optimization capabilities. The square is modeled with quadrilateral plate elements (CQUAD4) and is fixed at all four edges. The objective is to maximize the first frequency of the structure with a given bead dimension (minimum bead width = 10.0, maximum bead height = 20.0, draw angle = 70.0).

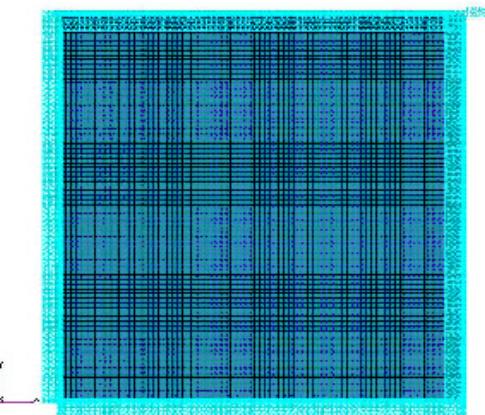


Figure 7-63 A Square

Input

The input data for this example is given in Listing 7-11. The Bulk Data entry BEADV=1 defines the topography designable region. It is noticed that element normals are used for bead vectors (draw direction) and all grids associated with the boundary condition are fixed during optimization. PARAM, POST, -1 outputs results for Patran.

Listing 7-11 Input File for Example 2

```
$Topography opt example one
SOL 200
CEND
TITLE = MSC Nastran job created on 28-Nov-07
ECHO = NONE
$ Direct Text Input for Global Case Control Data
DESOBJ(MAX) = 1
SUBCASE 1
$ Subcase name : Default
SUBTITLE=Default
SPC = 2
METHOD = 1
DISPLACEMENT(SORT1,REAL,PLOT)=ALL
ANALYSIS=MODES
BEGIN BULK
EIGRL,1,,,20
```



```
$ Direct Text Input for Bulk Data
$ Elements and Element Properties for region : ps1
$
$ BEADVAR, ID, TYPE, PID, MW, MH, ANG, BF, SKIP.
$
BEADVAR, 1, PSHELL, 1, 10., 20.0, 70.0, YES, BOTH
DRESP1, 1, MODES, FREQ,,,1
PARAM POST -1
```

Output

Figure 7-64 shows the topography optimized result by using Patran. The first frequency has increased from 0.568HZ at the initial design to 4.78 HZ.

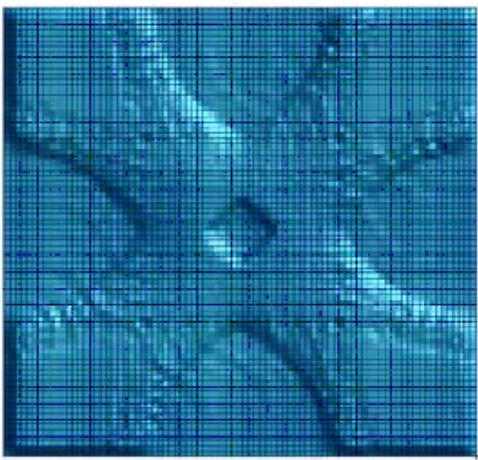


Figure 7-64



Postprocessing Topography Results

1. Run the Topography job. Create a new Patran database and read model and results from .op2 file.
2. Click **Results-> Action Create, Object Deformation**
3. Select the **Final Results Case**
4. Pick the desired case and Shape Change

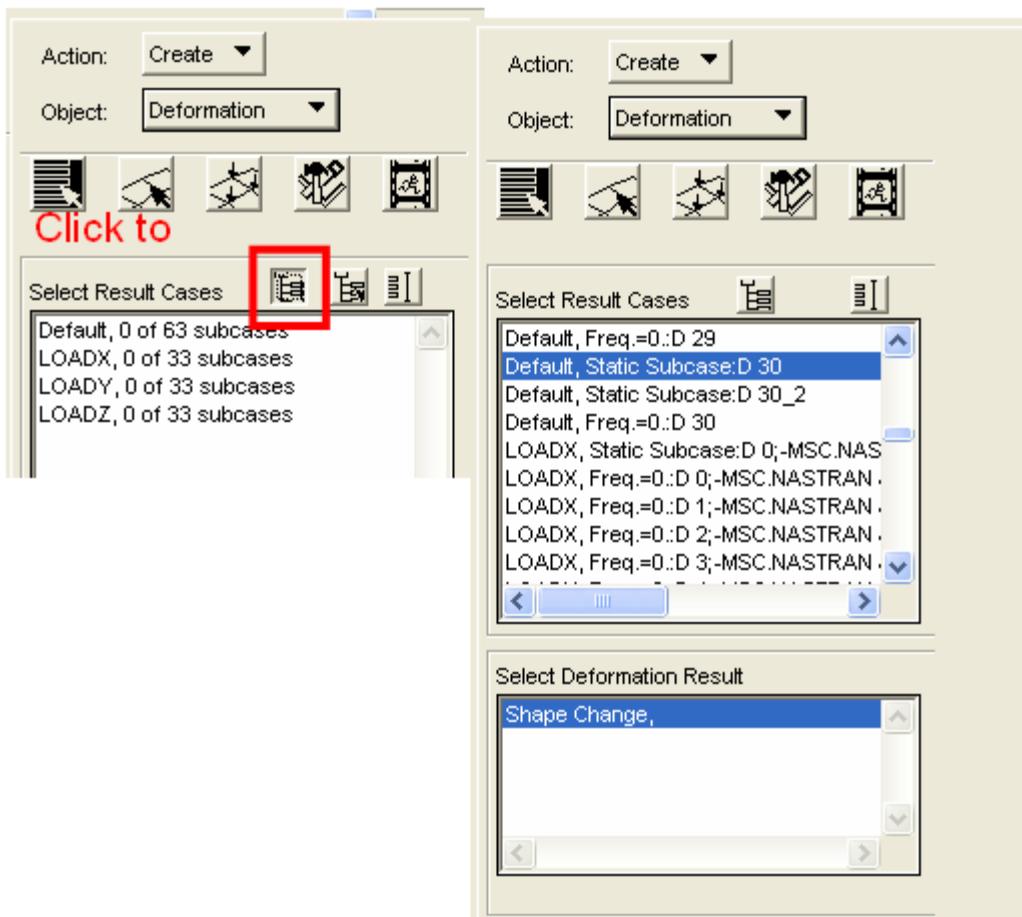


Figure 7-65 Selecting Shape Change

5. Modify the Display Attributes by setting deformed color to red, undeformed color to white and scale to True Scale.

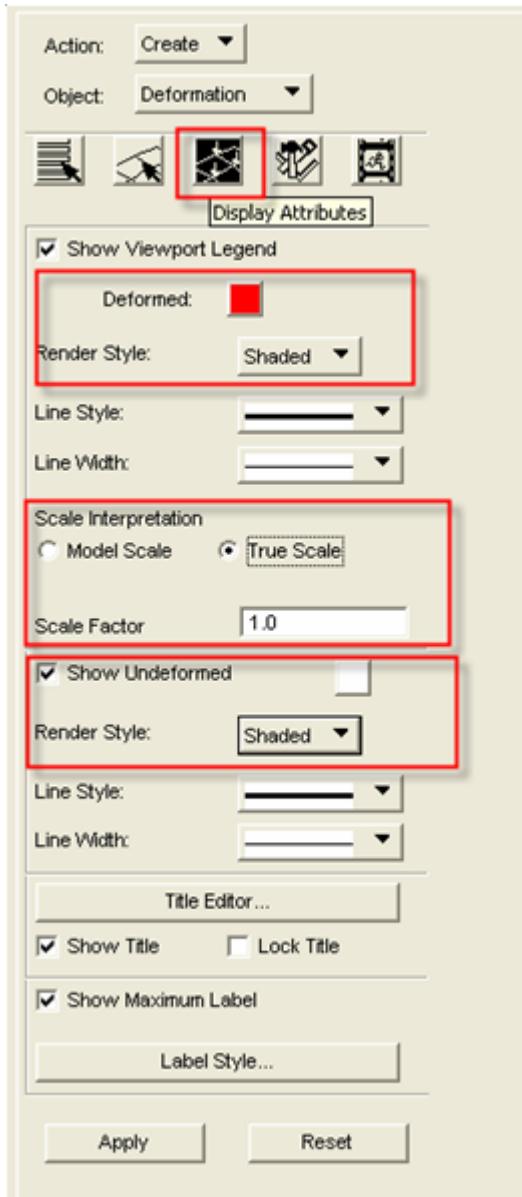


Figure 7-66 Setting Display Attributes

6. Result will show the original mesh (white) and optimization results in relief (red).



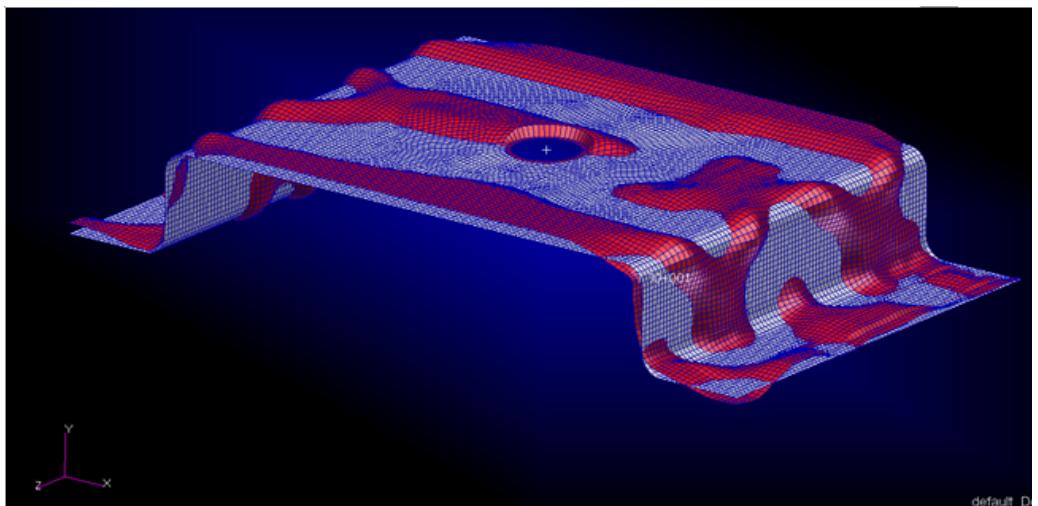


Figure 7-67 Optimized bead pattern

Toptimization

The Toptimize feature of Patran allows for the pre-processing of any Topology, Topometry or Topography job. This section illustrates setting up of Topometry and Topography jobs through examples.

Example 1 - Topography

A square plate is supported at the four corners and a point load acting normal to the plate is applied at its centre. The bead pattern is to be designed such that the compliance will be minimized subject to a constraint on the volume.

Follow the procedure below to pre-process the optimization problem in Patran. (It is assumed that the model, properties and load cases have already been created.)

Step 1: Select Toptomize

Click **Analysis**. On the Analysis form, select **Action**: Toptomize, **Object**: Entire Model, **Method**: Analysis Deck.



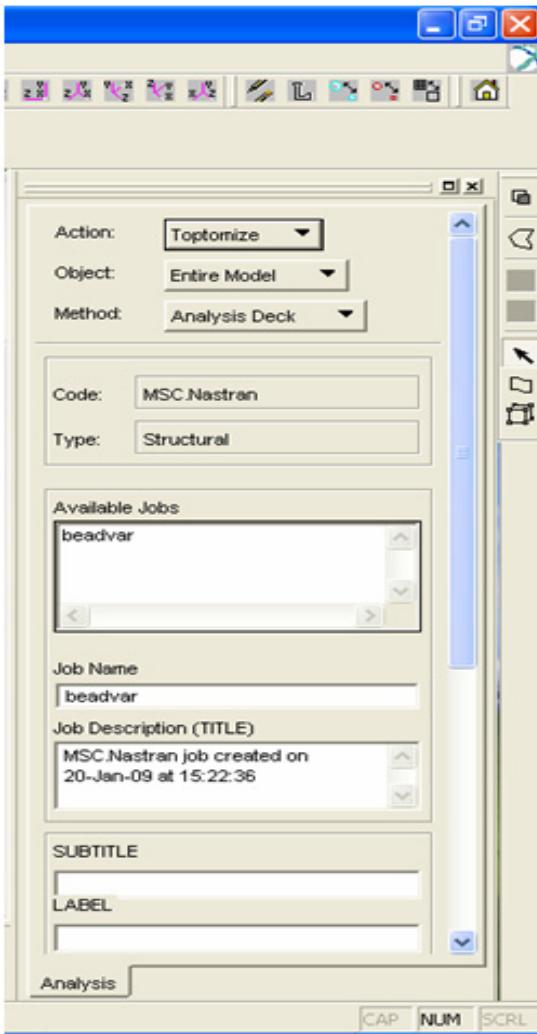


Figure 7-68 Initiating Toptomize for Topography Optimization

Step 2: Select Topography optimization, objective(s) and constraints

- a. On the Analysis form, click the **Objectives and Constraints** tab. On the Objectives and Constraints form, select **Type:** Topography.
- b. Select Objective Function(s) as Minimize Compliance or Maximize Frequency/Eigenvalue (both can also be selected if required). If Maximize Frequency/Eigenvalue is selected, then the desired mode number for which frequency is to be maximized needs to be entered in Mode Number(s) (comma or space separated). There is also the option for requesting Mode Tracking by checking the Track Modes check box. In this example, Minimize Compliance is selected.



- c. Frequency constraint can be set by entering the mode number and corresponding lower bound on frequency (the user can set multiple frequency constraints) in Frequency Constraint Target(s). (Not used in this example).
- d. Enter constraint on volume or weight in Constraint Target. (Volume is selected and target of 10 is entered in this example)
- e. Click **OK**.

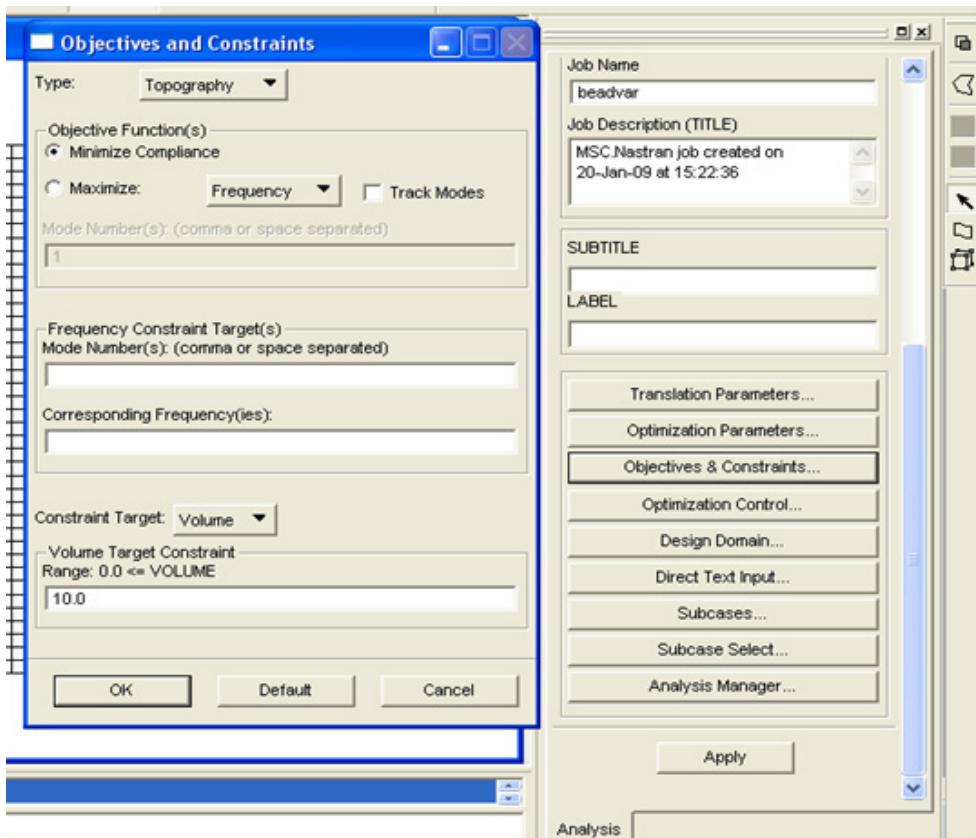


Figure 7-69 Creating objective and constraint

Step 3: Enter the fields of BEADVAR entry

- a. On the Analysis form, click on the **Optimization Control** tab.
- b. In the Optimization Control Parameters form, enter Lower Bound, Upper Bound, Minimum Bead Width, Maximum Bead Height and Draw Angle.
- c. Check the Buffer Zone and allow the default value on SKIP field by selecting Both on the Exclude from Design option.
- d. The user can also specify the values for maximum number of design cycles (DESMAX), move limit (DELXV) and convergence tolerance (CONV1).



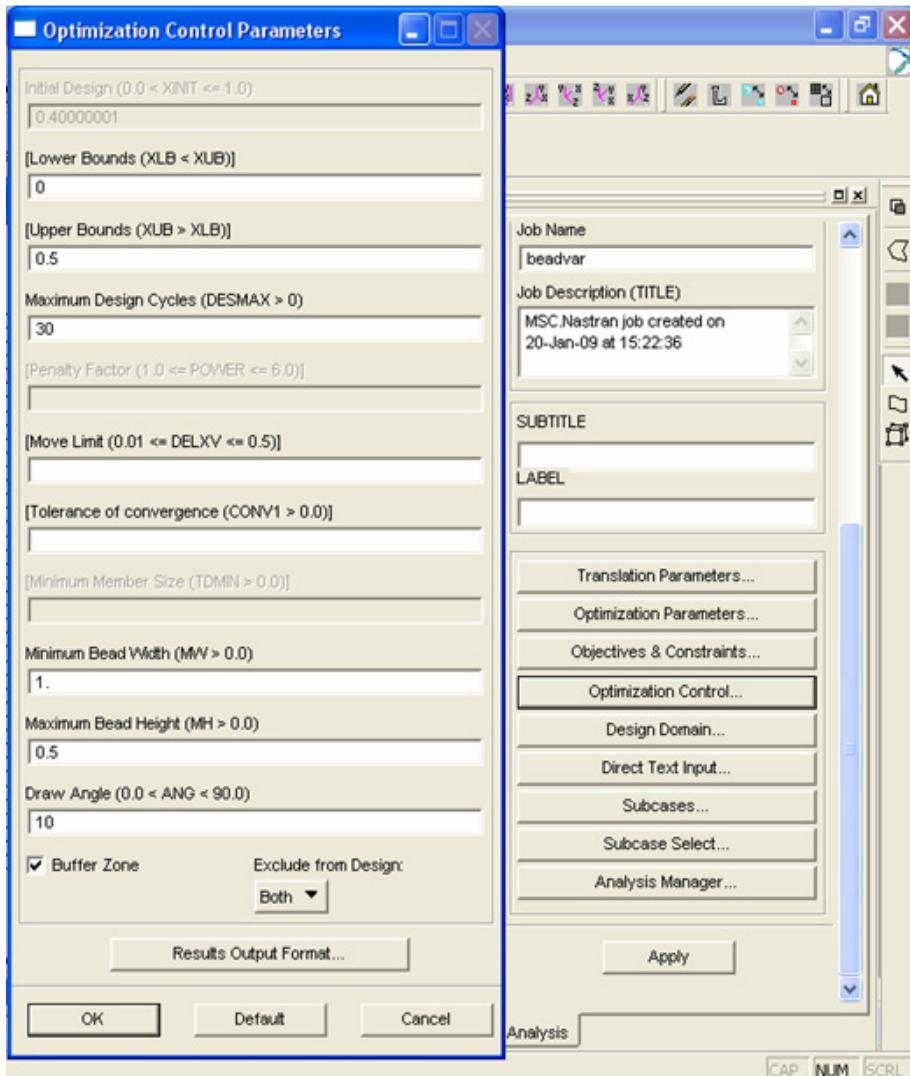


Figure 7-70 Creating the BEADVAR entry

Step 4: Select the Properties to be designed and enter the DESVAR and GRID lines of the BEADVAR entry.

- On the Analysis form, click the **Design Domain** tab.
- In the Design Domain form, click prop1 property in Valid Properties.
- Click the **Manufacturing Constraints** tab.
- In the Manufacturing Constraints form, select the **Normal** radio button in Extrusion direction.
- Nodes to be excluded from or included to the design can also be entered in the form.



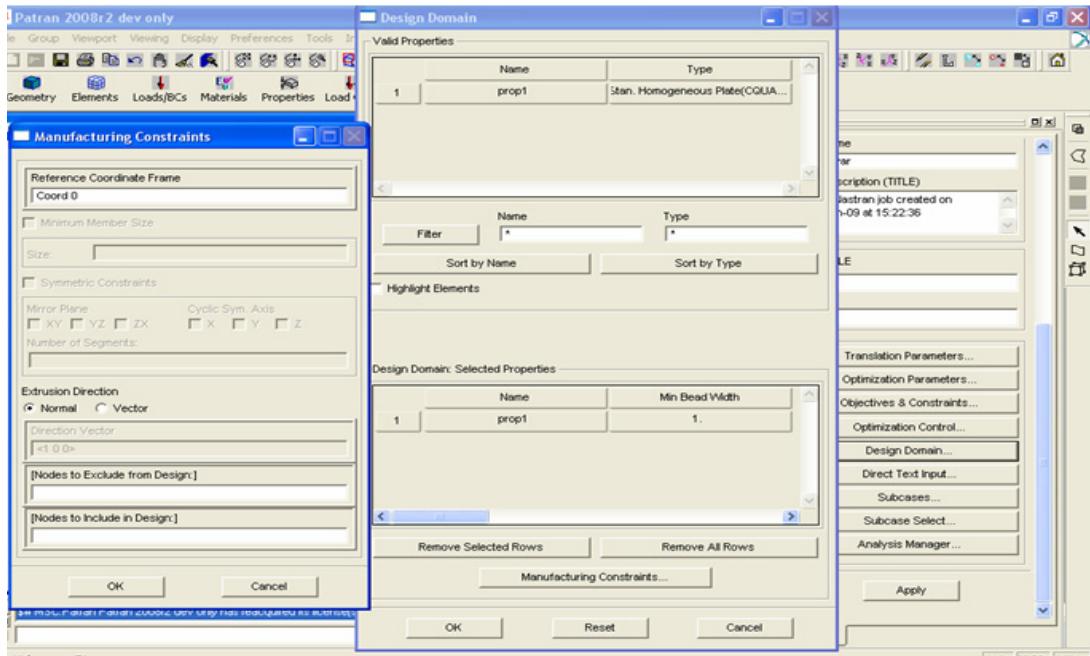


Figure 7-71 Manufacturing constraints

Step 5: Select the subcases

- On the Analysis form, click on the **Subcase Select** tab.
- In the Subcase Select form, select Solution Type 101 Linear Static. Select the created subcase from the Subcases Available list. (For composite objective such as sum of average compliance of static subcases and frequency of selected mode number in Step b, Solution Type 103 and available Normal Modes subcase can also be selected.)
- Click **OK**.
- Click **Apply** on Analysis form to generate the input deck.



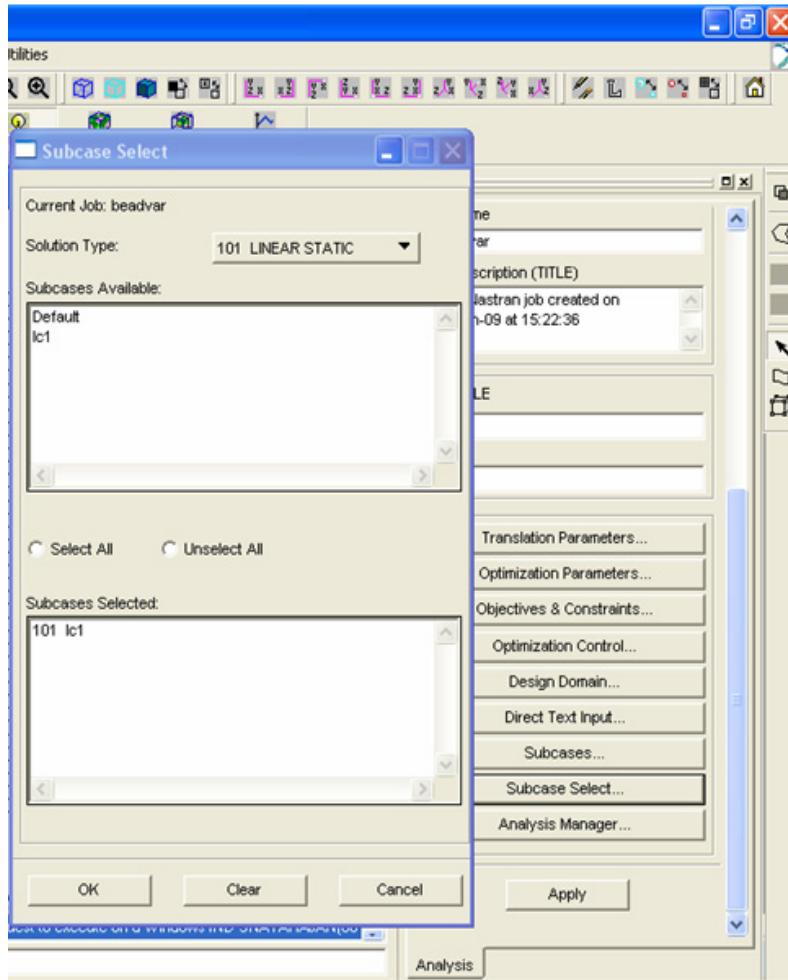


Figure 7-72 Selecting subcase

Example 2 - Topometry Optimization

The bridge problem is to be designed by continuous variation of shell thickness by using the TOMVAR entry.

Follow the procedure below to pre-process the optimization problem in Patran. (It is assumed that the model, properties and load cases have already been created.)

Step 1: Select Toptimize as in Step 1 of Example 1 above.

Step 2: Select Topometry Optimization and enter Objective Function and Constraints

- Click the **Objectives and Constraints** tab on the Analysis form.



- b. In the Objectives and Constraints form, select **Type:** Topometry.
- c. Select Minimize Compliance as the Objective Function(s).
- d. Select Mass Fraction as the Constraint Target and enter the value of the upper bound on fractional mass. (Also available are weight and volume constraints).
- e. Click **OK**.

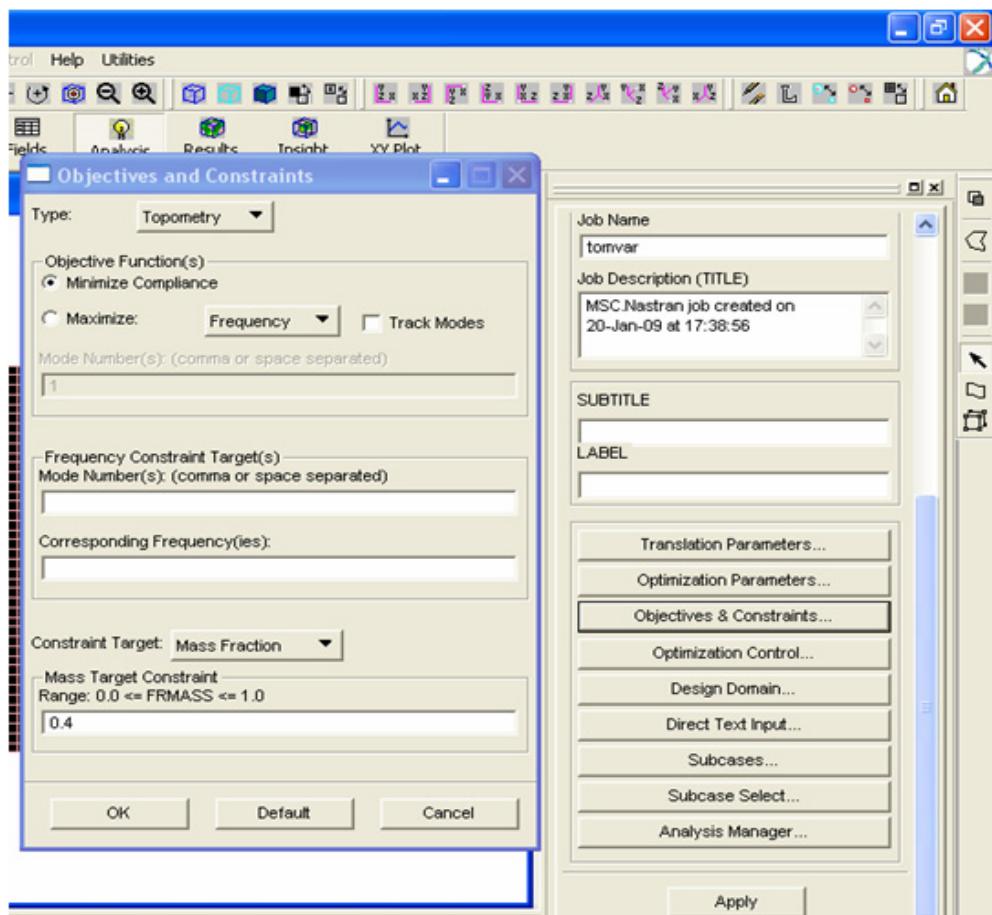


Figure 7-73 Toptimize for topometry optimization

Step 3: Selecting the property to be designed and entering the bounds on it

- a. On the Analysis form click on the **Optimization Control** tab.
- b. In the Optimization Control Parameters form, enter the values of XINIT, XLB, XUB, and DELXV.
- c. Select the property to be designed, In this example we select the Thickness.
- d. Click **OK**.



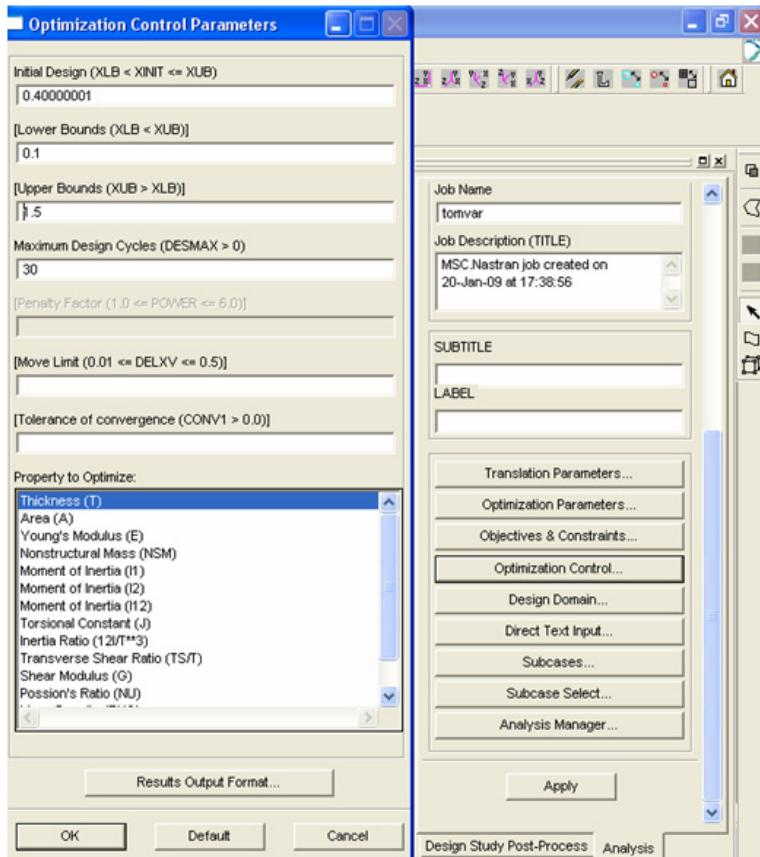


Figure 7-74 Property to Optimize

Step 4: Select the Design Domain

- On the Analysis form click on the **Design Domain** tab.
- In the Design Domain form, click on the prop1 property in Valid Properties. Then click **OK**.



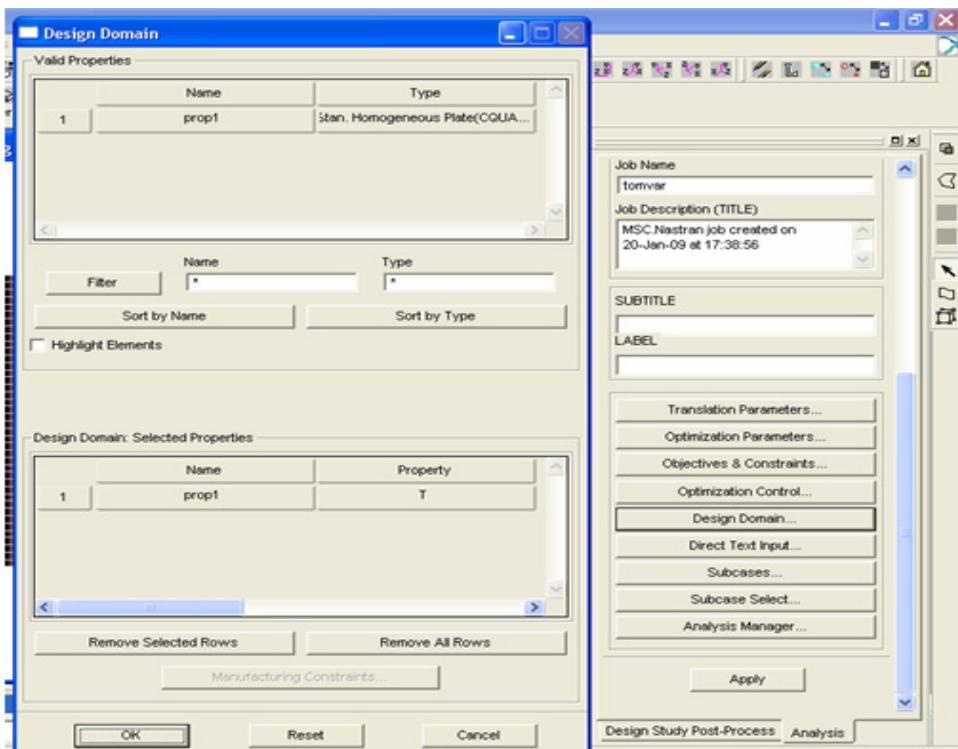


Figure 7-75 Design Domain

Step 5: Select the subcases (same as [Step 5:](#) in example 1 above).

This creates the deck for topometry optimization.



8

Example Problems

- Introduction 488
- Three-Bar Truss 490
- Vibration of a Cantilevered Beam (Turner's Problem) 513
- Cantilevered Plate 518
- Stiffened Plate 527
- Dynamic Response Optimization 533
- Twenty-Five Bar Truss, Superelement and Discrete Variable Optimization 544
- Design Optimization with Composite Materials with Fully Stressed Design 552
- Acoustic Optimization 559
- RMS Response 567
- Transient Dynamic Optimization 574
- External Response to Include Alternative Buckling Response 579
- Continuing the Design Process in a Subsequent Job 590
- Cantilevered Plate with Redundant Materials 598



Introduction

This chapter includes a wide range of examples intended to highlight and illustrate the Design Sensitivity and Optimization features of MSC Nastran. Though by no means exhaustive, these seventeen examples cover most major features including both Property and Shape Optimization, Dynamic Response Optimization, Superelements, Composites, Acoustic Responses, Discrete Variables, Fully Stressed Design, External Responses and Restarts. Additional examples can be found as given in Table 8-1:

Table 8-1 Additional Optimization Examples in MSC Nastran Guides

GUIDE	Chapter	Name	Description
Aeroelastic Analysis	10	Ha200a.dat	Sensitivity with aeroelastic analysis
	10	Ha200b.dat	Optimization with aeroelastic analysis
MSC Nastran Demonstration Problem Manual	29	Nug-29a.dat	Road response optimization with engine mount stiffness using modal frequency response
	29	Nug-29b.dat	Like nug-29a with modified objective and constraints
	29	Nug-29c.dat	Like nug-29a with analysis in the physical coordinates
	34	Nug-34a.dat	Topology optimization of MBB Beam. Basic compliance minimization
	34	Nug-34b.dat	Minimum member size
	34	Nug-34c.dat	Mirror symmetry constraints
	34	Nug-34d.dat	Extrusion constraints
	34	Nug-34e.dat	One die casting constraints
	34	Nug-34f.dat	Casting constraints with two dies
	35	Nug-35.dat	Engine mount topology optimization
	36	Nug-36.dat	Topology optimization of a wheel with cyclic symmetry constraints

Running these examples on your own machine is a good way to familiarize yourself with some of the capabilities in MSC Nastran. The iterative nature of optimization is such that different versions of MSC Nastran and different computer platforms could result in different results from those shown here. It is expected that these differences will be small. All of these files are available in a directory delivered with your system. The *MSC_DOC_DIR/doc*/desopt directory contains these files with the extension of .dat (mscxxxx represents your version of MSC Nastran - replace the xxxx with your specific version). If you are unable to find these files, you should contact your System Administrator for assistance. It is



recommended you copy any example problem to your local directory, so you don't inadvertently create files in the delivery directory.

Another good way to learn is by modifying these files. You might try adding Shape variables to a Property Optimization example, adding analysis disciplines, modifying the constraints, and so on. Most of these files 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.

Note: The results shown in this guide were obtained from MSC Nastran 2001, 2004, 2010 and 2014.1 running on Linux and Windows platforms. Results from running these cases on different platforms or versions of MSC Nastran may result in somewhat different results, but they should be similar.



Three-Bar Truss

A common task in design optimization is to reduce the mass of a structure subjected to several load conditions. Figure 8-1 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.

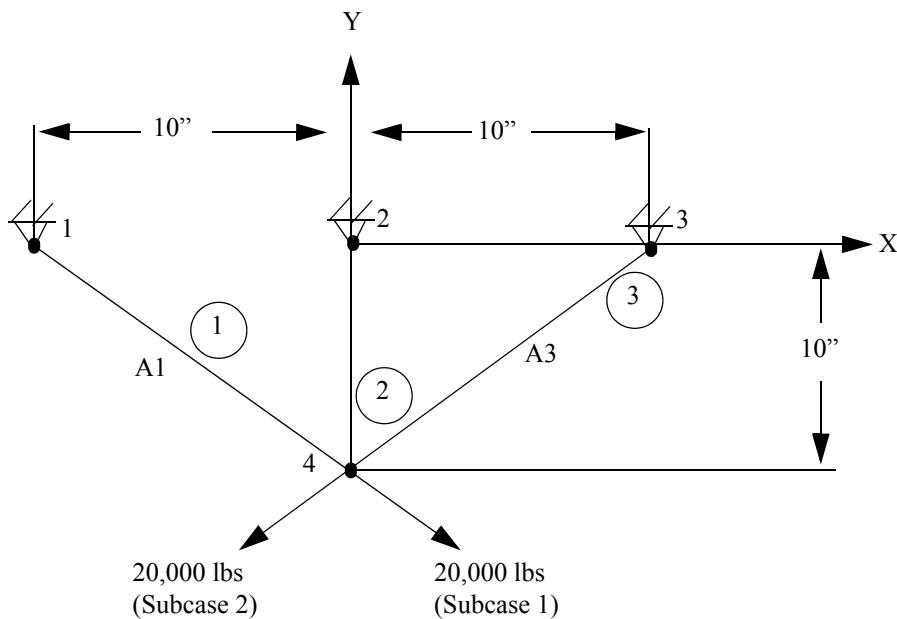


Figure 8-1 Three-Bar Truss

An important, but often overlooked consideration is that the optimization capability in MSC 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 or buckling 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.)



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: $E = 1.0E7 \text{ psi}$

Weight density = 0.1 lbs/in³

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

Displacement: ± 0.2 at grid 4 in x end of y directions

The input data for this problem is given in [Listing 8-1](#). 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.

Listing 8-1 Input File for DSOUG1

```
ID MSC  DSOUG1 $  
TIME 10      $  
SOL 200      $  OPTIMIZATION  
CEND  
TITLE = SYMMETRIC THREE BAR TRUSS DESIGN OPTIMIZATION -      DSOUG1  
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
SPC1   100     123456  1      THRU   3
$ 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      NAME    PMIN    PMAX    CO      +
$+      DVID1      COEF1   DVID2   COEF2   ...
DVPREL1 10      PROD    11      A       1       1.0
1       1.0
DVPREL1 20      PROD    12      A       2       1.0
2       1.0
DVPREL1 30      PROD    13      A       3       1.0
3       1.0
$
$...STRUCTURAL RESPONSE IDENTIFICATION
$DRESP1 ID      LABEL   RTYPE   PTYPE   REGION  ATTA    ATTB    ATT1    +
$+      ATT2      ...
DRESP1 20      W       WEIGHT
DRESP1 21      U4      DISP
DRESP1 23      S1      STRESS  PROD
12
12      13
13
$
$...CONSTRAINTS
$DCONSTR DCID   RID      LALLOW  UALLOW
DCONSTR 21      21     -0.20   0.20
DCONSTR 21      23    -15000. 20000.
$
$...OPTIMIZATION CONTROL:
$-
DOPTPRM DESMAX 10      DELP    0.5     P1      1       P2      15
$-
$.....2.....3.....4.....5.....6.....7.....8.....9.....0
ENDDATA

```

Turning to the design model description, we see three design variables are used to control the individual rod element cross-sectional areas. The set of DESVAR, DVPREL1 entries define the relations:

$$A_i = 1.0X_i \quad i = 1, 2, 3 \quad (8-1)$$



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 is symmetric and the two loading cases act to impose the same critical design loads on each side of the plane of symmetry, we expect the optimizer to yield a symmetric final design with $A1 = A3$. We may want to enforce some type of symmetry ourselves, perhaps to address ease-of-manufacture concerns. This can be done using the DLINK entry to explicitly link design variables together. Although strictly not necessary, DLINK 1 is used to define the relation

$$X_3 = 1.0X_1 \quad (8-2)$$

The value of X_3 now depends on X_1 . The optimizer now 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. These cost savings can be significant in a larger problem.

An alternate method of linking would have been placing rod elements 11 and 13 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. Alternatively, you may wish to remove the DLINK entry to assure yourself that MSC Nastran will produce a symmetric design in this case even when the design model allows for $X_1 \neq X_3$.

DRESP1 entries define the design responses: displacement and stress responses which 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 encountered, 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. As was discussed in [Move Limits](#), 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. 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 six iterations. A total of seven 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 ($W = L_1X_1 + L_2X_2 + L_3X_3$), 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. 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.



Listing 8-2 Selected Output for DSOUG1

```
***** NORMAL CONVERGENCE CRITERIA SATISFIED ***** (HARD CONVERGENCE DECISION LOGIC)
*****
CONVERGENCE ACHIEVED BASED ON THE FOLLOWING CRITERIA
(HARD CONVERGENCE DECISION LOGIC)
RELATIVE CHANGE IN OBJECTIVE      9.9111E-04 MUST BE LESS THAN 1.0000E-03
OR      ABSOLUTE CHANGE IN OBJECTIVE      2.6741E-03 MUST BE LESS THAN 1.0000E-20
      --- AND ---
MAXIMUM CONSTRAINT VALUE      2.4029E-03 MUST BE LESS THAN 5.0000E-03
      (CONVERGENCE TO A FEASIBLE DESIGN)
      --- OR ---
MAXIMUM OF RELATIVE PROP. CHANGES   6.1176E-02 MUST BE LESS THAN 1.0000E-03
AND      MAXIMUM OF RELATIVE D.V. CHANGES   6.1176E-02 MUST BE LESS THAN 1.0000E-03
      (CONVERGENCE TO A BEST COMPROMISE INFEASIBLE DESIGN)
*****
```

```
*****
SUMMARY OF DESIGN CYCLE HISTORY
*****
```

(HARD CONVERGENCE ACHIEVED)

NUMBER OF FINITE ELEMENT ANALYSES COMPLETED	8
NUMBER OF OPTIMIZATIONS W.R.T. APPROXIMATE MODELS	7

OBJECTIVE AND MAXIMUM CONSTRAINT HISTORY

CYCLE NUMBER	OBJECTIVE FROM APPROXIMATE OPTIMIZATION	OBJECTIVE FROM EXACT ANALYSIS	FRACTIONAL ERROR OF APPROXIMATION	MAXIMUM VALUE OF CONSTRAINT
INITIAL		4.828427E+00		-3.234952E-01
1	3.012648E+00	3.012949E+00	-9.994284E-05	-5.576856E-03
2	2.822569E+00	2.822634E+00	-2.289048E-05	-1.737715E-02
3	2.735552E+00	2.735578E+00	-9.325558E-06	-3.660352E-03
4	2.711009E+00	2.711009E+00	8.794460E-08	-3.000000E-04
5	2.702893E+00	2.702891E+00	9.702960E-07	1.269531E-05
6	2.698114E+00	2.698095E+00	7.245974E-06	9.239258E-04
7	2.695410E+00	2.695421E+00	-3.891941E-06	2.402930E-03

1 SYMMETRIC THREE BAR TRUSS DESIGN OPTIMIZATION - DSOUG1 SEPTEMBER 7, 2009 MSC NASTRAN 9/ 4/09
PAGE 80
BASELINE - 2 CROSS SECTIONAL AREAS AS DESIGN VARIABLES
0

DESIGN VARIABLE HISTORY

INTERNAL EXTERNAL	DV. ID. DV. ID.	LABEL	INITIAL : 1	: 2	: 3	: 4	: 5
1 1 A1 1.0000E+00 : 7.1168E-01 : 7.7885E-01 : 7.9878E-01 : 8.1423E-01 : 8.2945E-01							
2 2 A2 2.0000E+00 : 1.0000E+00 : 6.1972E-01 : 4.7627E-01 : 4.0803E-01 : 3.5686E-01							
3 3 A3 1.0000E+00 : 7.1168E-01 : 7.7885E-01 : 7.9878E-01 : 8.1423E-01 : 8.2945E-01							

INTERNAL EXTERNAL	DV. ID. DV. ID.	LABEL	6 : 7	: 8	: 9	: 10	: 11
1 1 A1 8.3904E-01 : 8.3107E-01 :							
2 2 A2 3.2493E-01 : 3.4481E-01 :							
3 3 A3 8.3904E-01 : 8.3107E-01 :							

*** USER INFORMATION MESSAGE 6464 (DOM12E)
RUN TERMINATED DUE TO HARD CONVERGENCE TO AN OPTIMUM AT CYCLE NUMBER = 7.

We will now see how the input deck for this example can be created in Patran.



Step 1: Open the Pre-Process Form

- a. Click Tools→Design Study→ Pre-Process

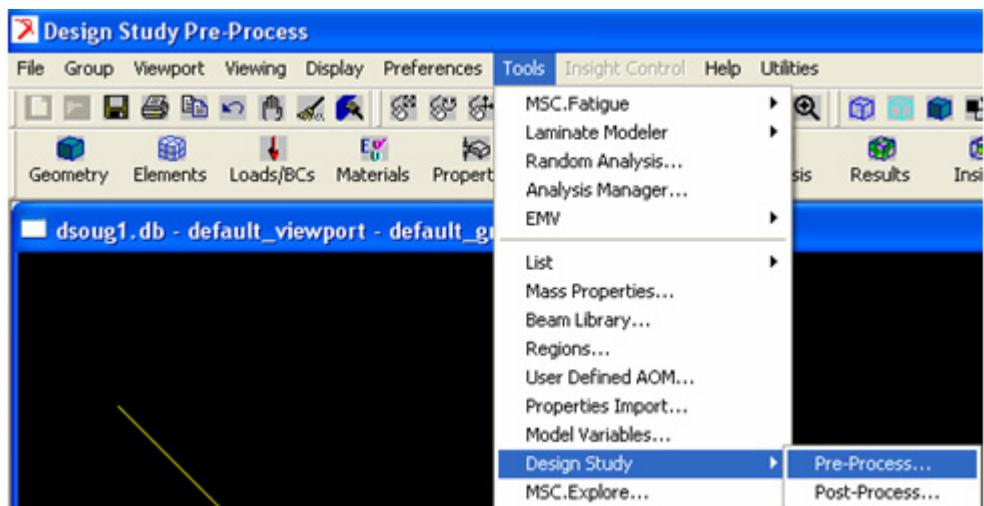


Figure 8-2 Initiating Pre-process

Step 2: Create Design Variables

Since we are designing the area which is a property on the PROD entry, we can create design variables easily by Property.

- a. On the Pre-Process form, select **Action**: Create, **Object**: Design Variable, **Type**: Property.
- b. Select **Dimension**: 1D and **Type**: Rod.



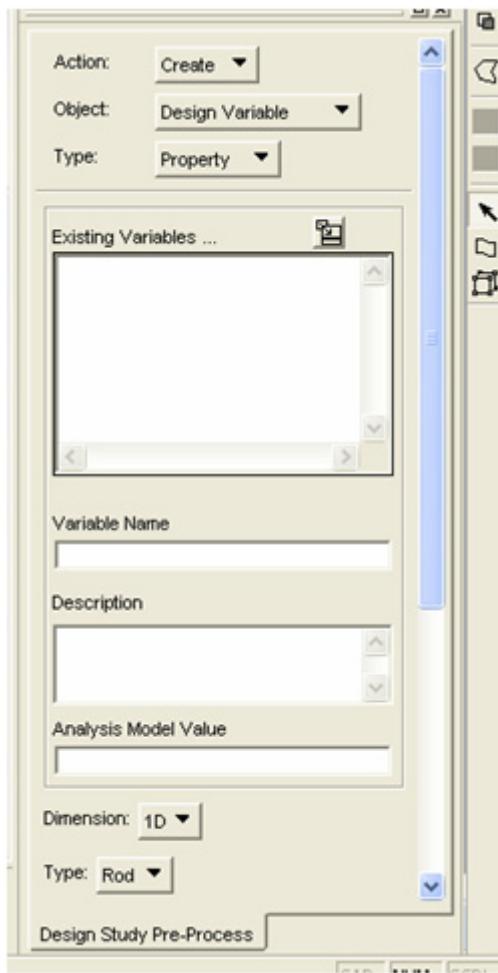


Figure 8-3 Creating Design Variables

- c. Then click the **Select Property Set(s)** icon and select prod1, click Select Property icon and select Area.
- d. Select Input Bounds as Lower/Upper and Value and enter the values 0.1 and 100.0 for Lower (L) and Upper (U), respectively.
- e. Click **Apply**.
- f. Similarly create design variables for area of prod2 and prod3.



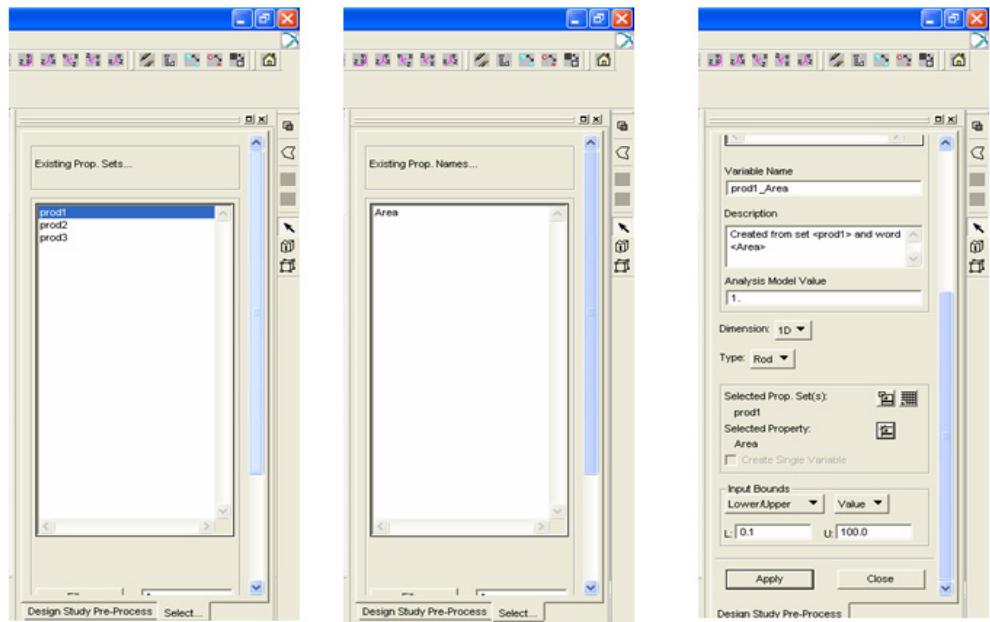


Figure 8-4 Setting Bounds on Design Variables

Step 3: Create relation between design variables (area of prod3 = area of prod1)

- a. Select **Action**: Create, **Object**: Variable Relation, **Type**: General (DLINK)
- b. Enter 0.0 for Constant Term (C0) and 1.0 for Multiplier (CMULT).
- c. Select prod3_Area in Dependent Design Variable list.
- d. Click Define Indep Terms and select prod1_Area in Select Design Variable (IDVi) list on the General (DLINK) Independent Terms form.
- e. Click **OK** and **Apply**.



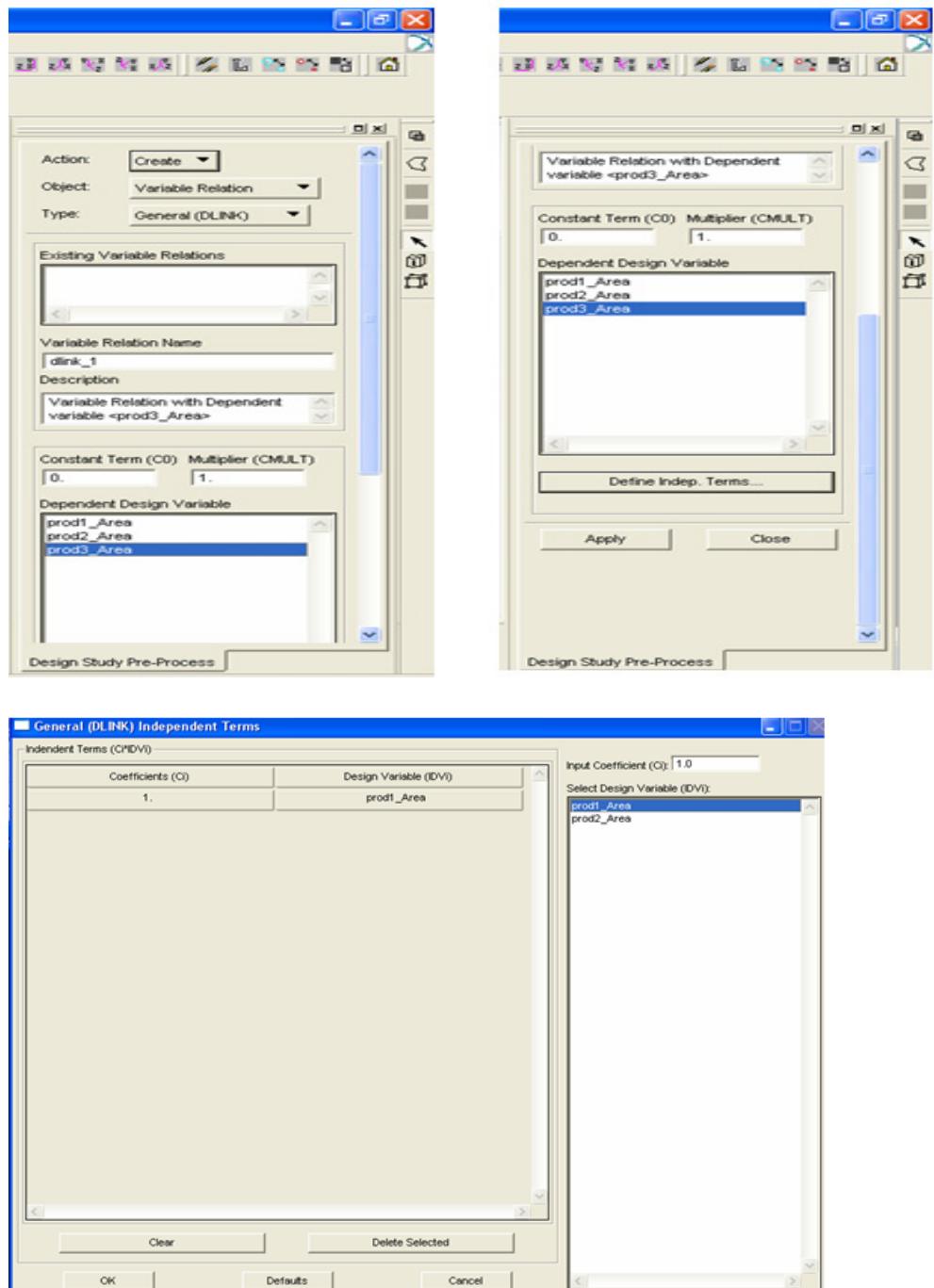


Figure 8-5 Creating variable linking: DLINK

Relation between design variables can also be set very easily, in this case, by selecting Type: Property Relation. Simply select the Dependent Property Set as prod3 and Property Name as Area, and Independent Property Set as prod1 and Property Name as Area. (Note that design variables prod1_Area and prod3_Area need not be defined prior to this step because the two design variables and their relation are automatically created.)

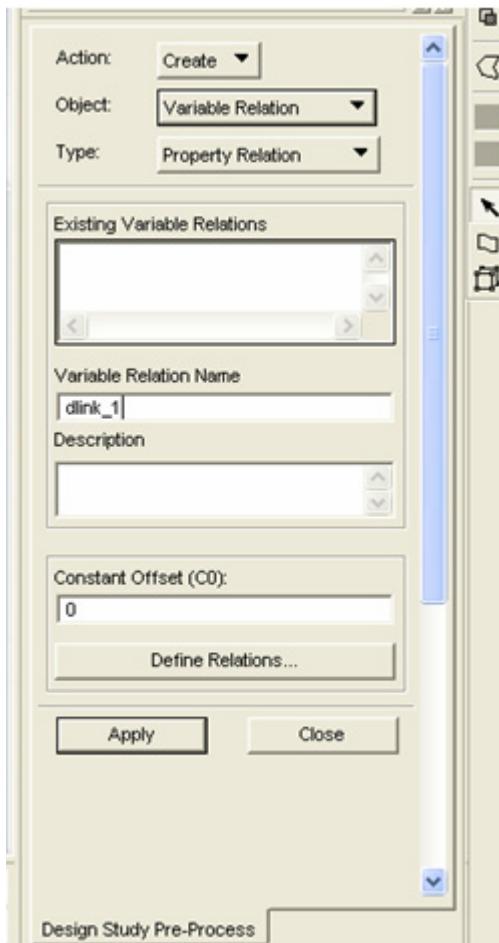


Figure 8-6 Creating variable linking: Property relation



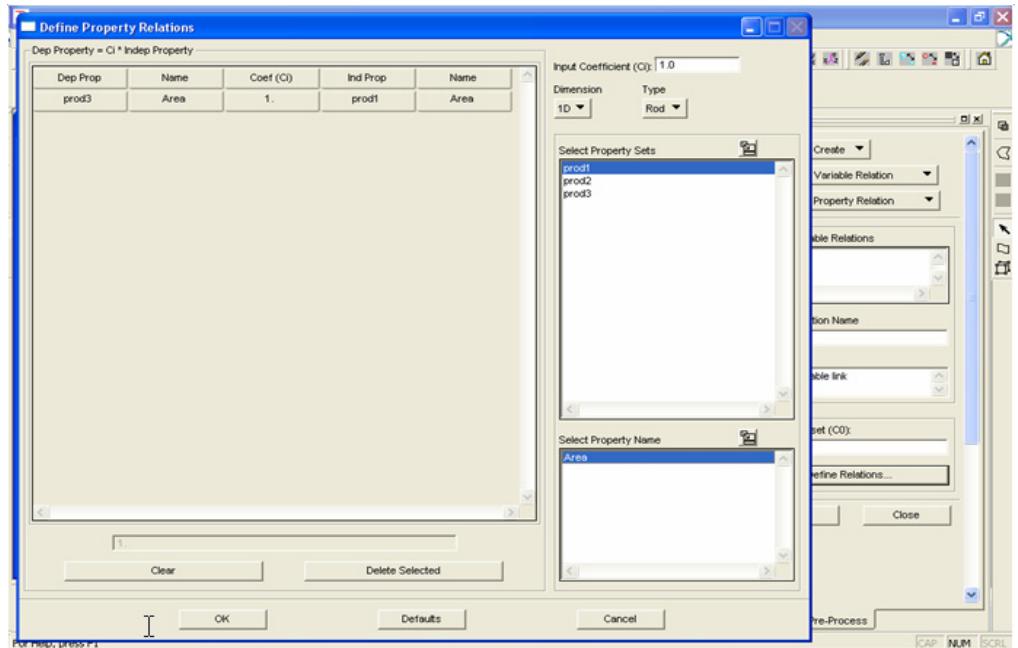


Figure 8-7 Define Property Relation

Step 4: Create Objective

- Select **Action**: Create, **Object**: Objective, **Solution**: Global, **Response**: Weight and **Min/Max**: Minimize.
- Enter weight in Objective Name and click **Apply**.



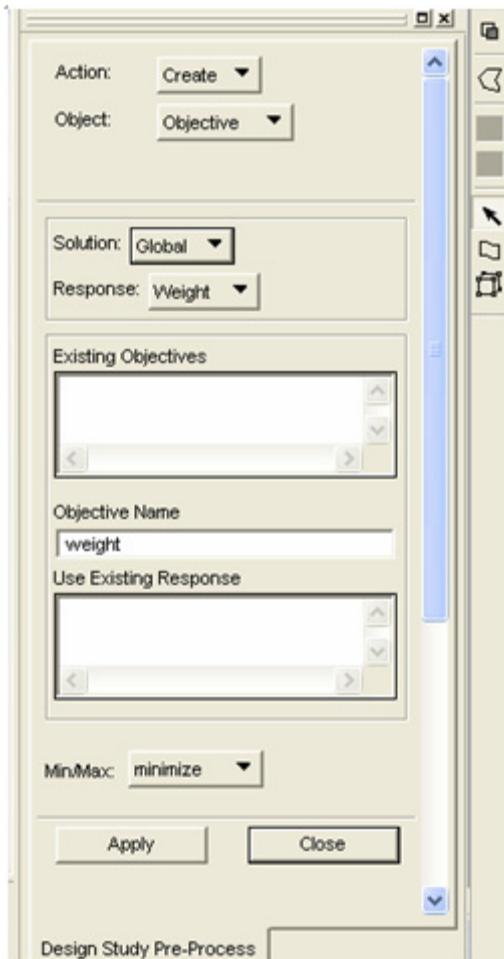


Figure 8-8 Creating Objective

Step 5: Create Displacement and Stress Constraints

- Select **Action**: Create, **Object**: Constraint, **Solution**: Linear Static, **Response**: Displacement
- Enter `disp_u_4` for Constraint Name, select FEM radio button for Constraint and pick node 4 to Select Node.
- Select TX for Displacement Component (another constraint has to be created for TY because Patran does not allow selection of more than one component) and enter values -0.2 and 0.2 for Lower Bound and Upper Bound respectively.
- Click **Apply**.
- For the stress constraints, repeat step 1, but select Stress for the Response.



- f. Enter sigma_123 for Constraint Name, select P_Set for Constraint Region (by Property) and select all properties in Select Existing Properties list.
- g. Select Axial for Stress Component and enter values -15000 and 20000 for Lower Bound and Upper Bound, respectively.

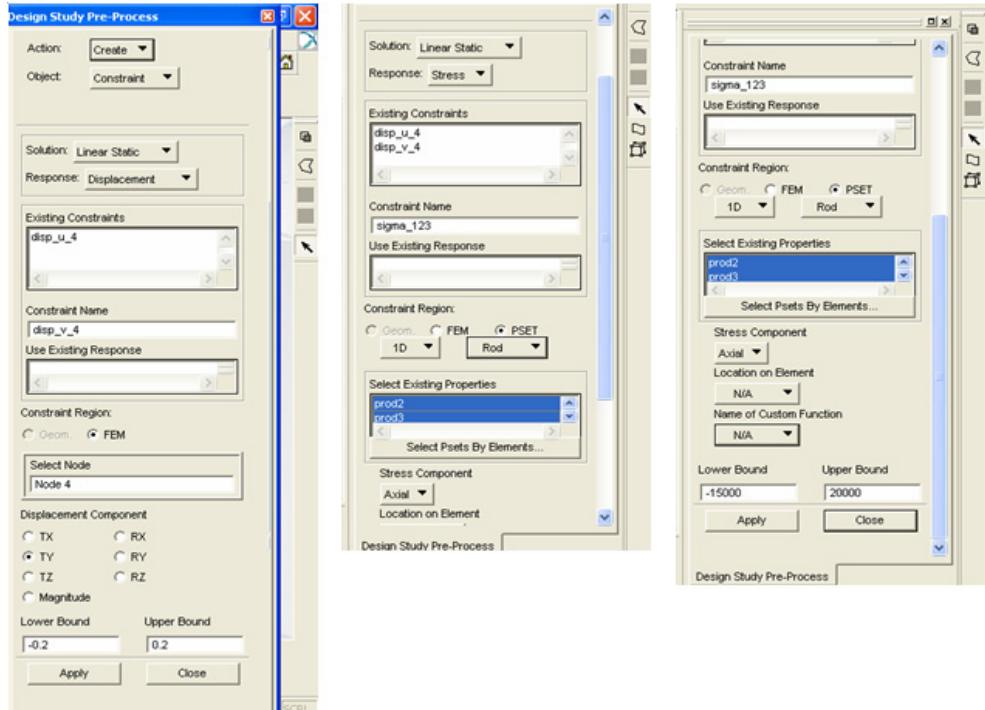


Figure 8-9 Creating constraints

We can also add a normal modes constraint. While it is not required for this example, we will take up this step to illustrate how multidisciplinary optimization decks can be set up using Patran.

Step 6: Creating normal modes constraint

- a. Select **Action: Create**, **Object: Constraint**, **Solution: Normal Modes**, **Response: Frequency**.
- b. Enter mode1 for Constraint Name, 1 for Frequency Mode Number and 10 for Upper Bound.
- c. Click **Apply**.



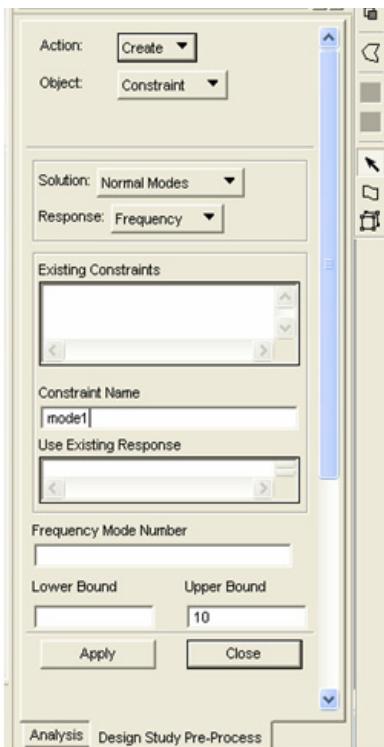


Figure 8-10 Creating normal modes constraint

Since the stress and displacement constraints will be considered for both the static load cases and normal modes constraint will be considered only for the normal modes subcase, we need to create constraint sets which will be selected for the corresponding subcases.

Step 7: Creating the static and modes constraint sets

- Select **Action**: Create, **Object**: Constraint Set, **Solution**: Linear Static.
- Enter staticset in Constraint Set Name, select all the constraints in the Constraints to be Included list (note that only the static case constraints are listed). Click **Apply**.
- Now change the Solution to Normal Modes, enter modeset in Constraint Set Name, select the constraint mode1 in the Constraints to be Included list. Click **Apply**.



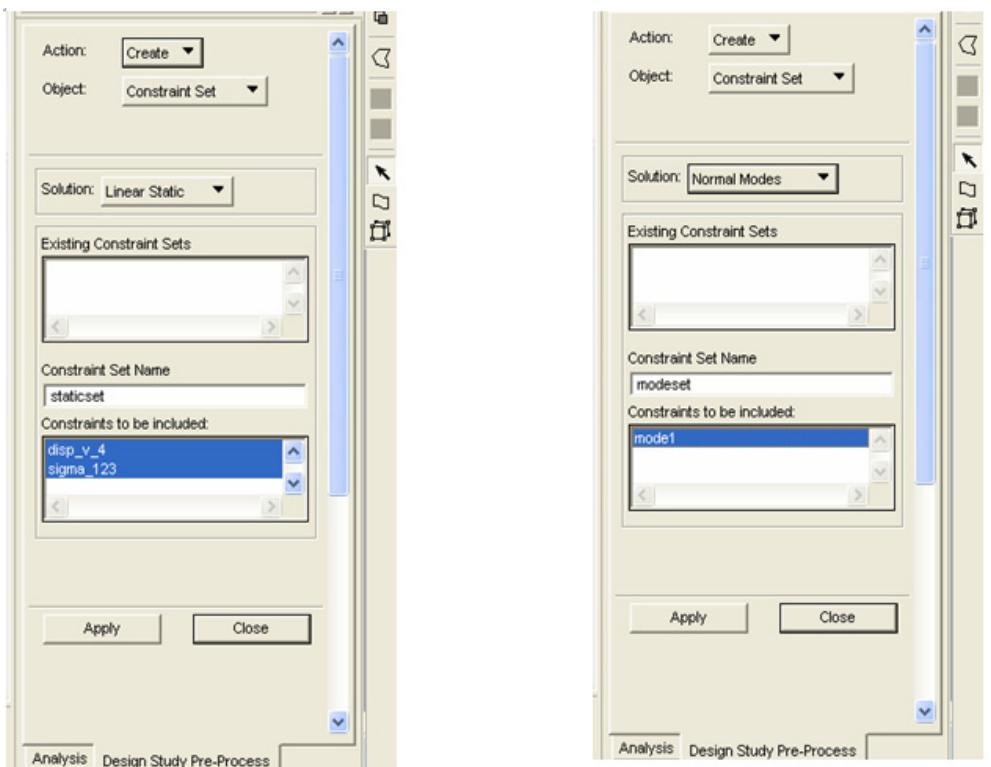


Figure 8-11 Creating Constraint Sets

Step 8: Create a Design Study

- a. Select **Action: Create**, **Object: Design Study** and enter ds1 for Design Study Name.
- b. In Design Study Setup, click the **Select Design Variables** tab.
- c. Enter Design Values (it is equal to the Analysis Values in this case) and lower bounds as 0.1 and upper bounds as 100 for each of the three design variables in the table.
- d. Click the **Select Objective** tab. Click the **Select All** radio button. Click **Close**.
- e. Repeat step 4 for Select Constraint Sets and Select Variable Relations.
- f. Click **Apply**.



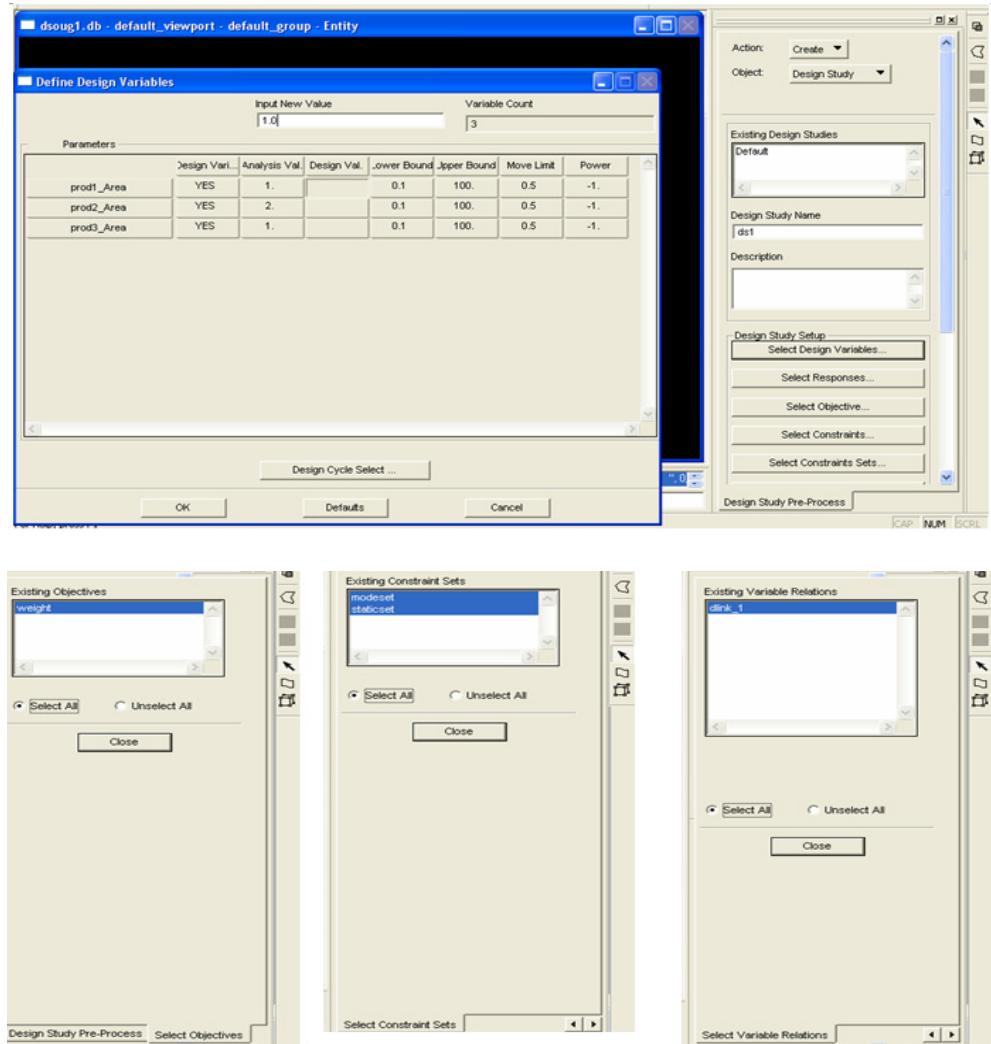


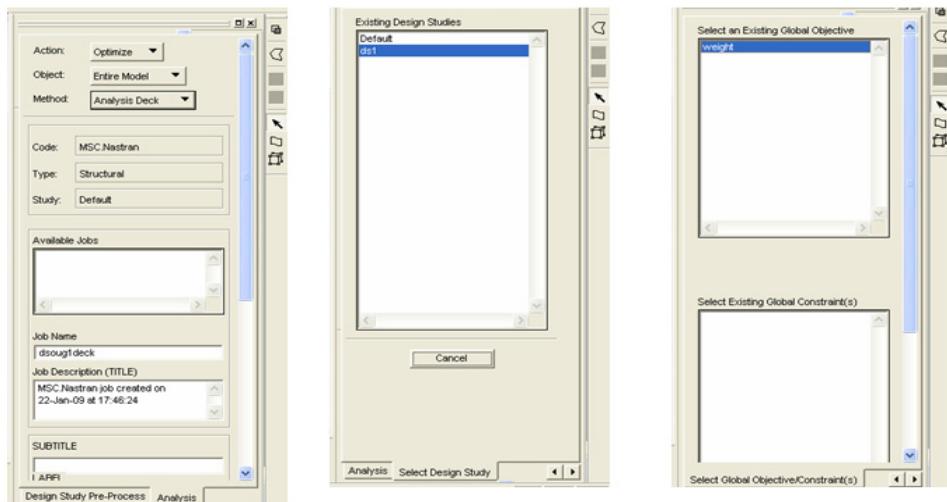
Figure 8-12 Creating Design Study

Step 9: Creating Subcases and overwriting default Optimization Parameters

- Click **Analysis**. Select **Action: Optimize**, **Object: Entire Model**, **Method: Analysis Deck**.
- Click the **Design Study Select** tab and select ds1.
- Click the **Global Obj./Constr. Select** tab and select weight.
- Click the **Optimization Parameters** tab. In the Optimization Parameters form, change DESMAX to 10 (default is 5), check Print Objective and Design Variables, Print Properties, Print All Constraints and Print All Responses (this is equivalent to P2 =15 on DOPTPRM entry).



- e. Change the value of P1 to 1 (default is 0).
- f. Click the **Advanced Optimization Parameters** tab.
- g. In the Advanced Optimization Parameters form, change the DELP value to 0.5 (default is 0.2). Click **OK**.



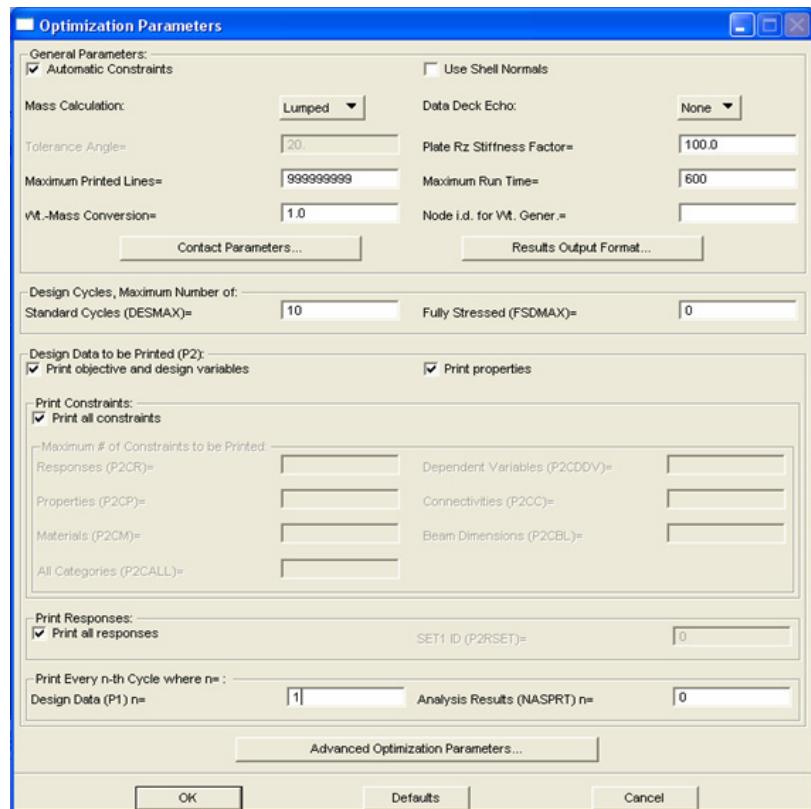


Figure 8-13 Setting Optimization Parameters



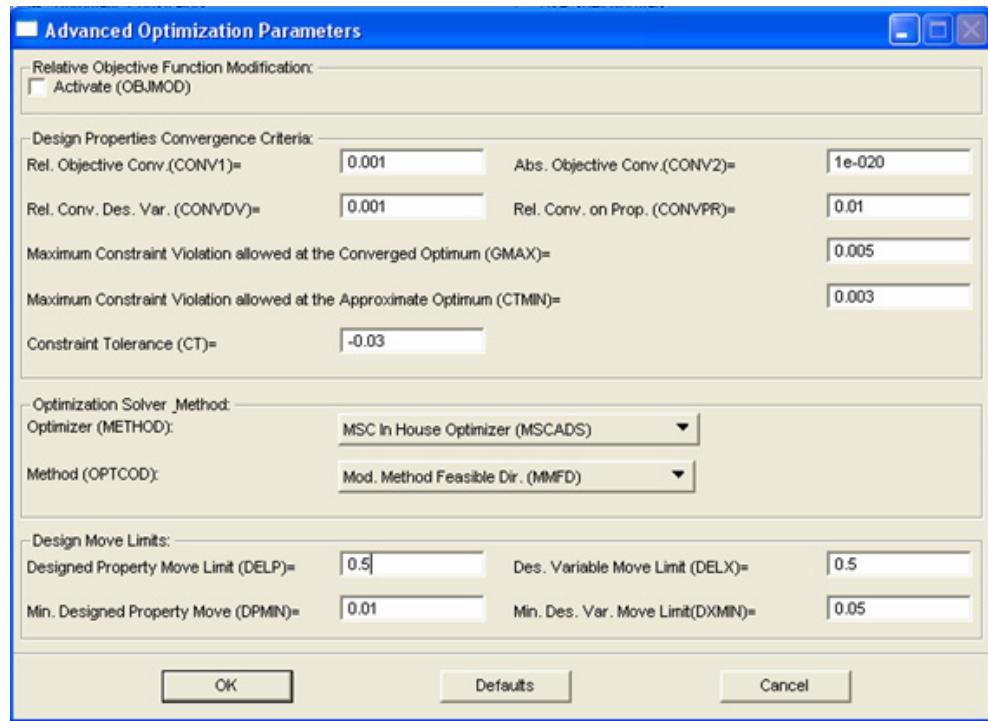


Figure 8-14 Setting up objectives and constraints

Now the two static and one normal modes subcases will be created

- h. Click the **Subcase** tab.
- i. In Subcase Create form, select Solution Type: 101 LINEAR STATIC.
- j. From Available Subcases, select lc1. Click the **Constraints/Objective** tab.
- k. In Set Up Constraints/Objective form, select **Type:** Constraints/Objective. Select Constraint Sets radio button and select **staticset** from Select Existing Constraint Sets list.
- l. Click **OK** and **Apply**.
- m. Similarly create lc2 subcase.
- n. To create the normal modes subcase, select Solution Type: 103 NORMAL MODES and select modeset from Select Existing Constraint Sets list.
- o. Click Subcase Parameters tab and fill up the form.



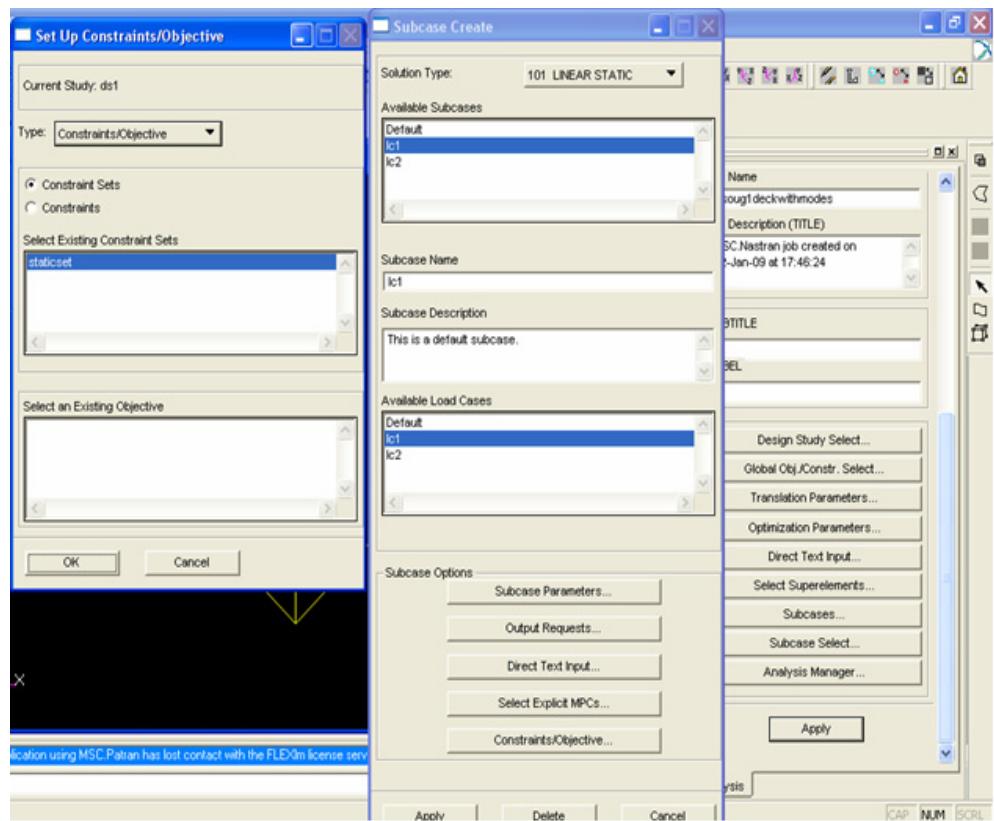


Figure 8-15 Subcase Parameters



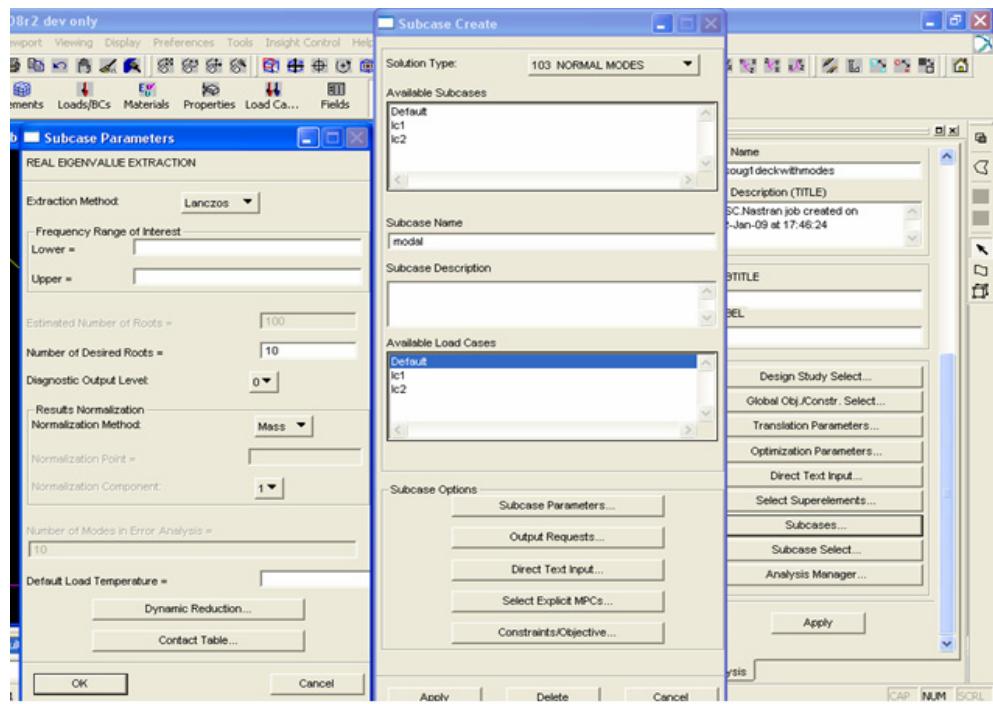
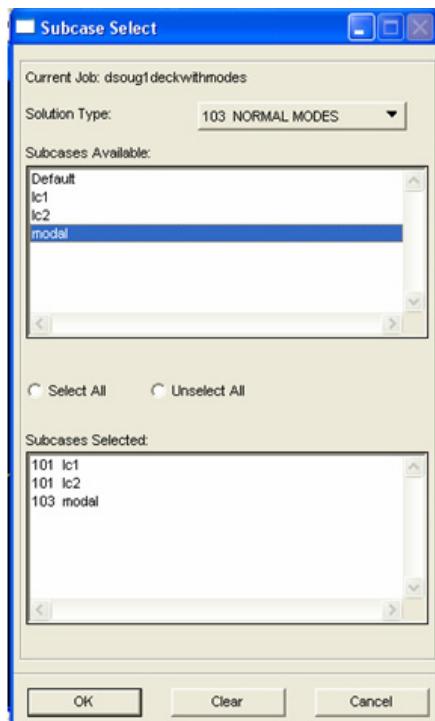


Figure 8-16 Selecting Subcases

- p. Click Subcase Select tab and select the two static subcases and one modal subcase just created.
- q. Click **Apply** to get the SOL 200 deck.





Vibration of a Cantilevered Beam (Turner's Problem)

This problem was originally published by M.J. Turner (see Reference 13.). The problem is to design a minimum weight structure while constraining the fundamental natural frequency to be at or above 20 Hz. The beam is symmetric about $Z = 0$ 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.

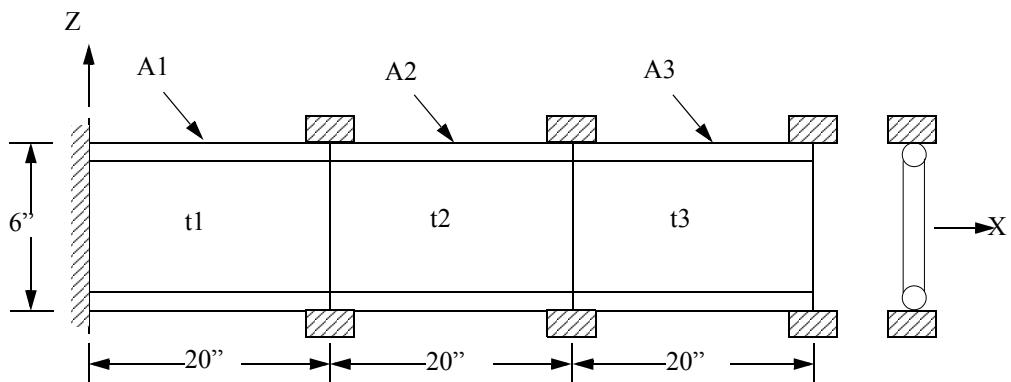


Figure 8-17 Cantilever Beam Vibration Model



Analysis Model Description

Web is modeled by QUAD4 elements

Caps are modeled by ROD elements

Material: $E = 1.03E7 \text{ psi}$

Poisson ratio = 0.3

Weight density = 0.1 lb/in³

Nonstructural 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 rod 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 8-3](#) for details.

Listing 8-3 Input File for DSOUG2

```
ID MSC DSOUG2 $ v2004 ehj 25-Jun-2003
TIME 10
SOL 200      $ OPTIMIZATION
CEND
TITLE      = VIBRATION OF A BEAM.                               DSOUG2
SUBTITLE   = TURNER'S PROBLEM
ECHO       = UNSORT
DISP       = ALL
STRESS     = ALL
METHOD     = 1
ANALYSIS   = MODES
DESOBJ(MIN) = 10    $ 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      1.0     0.001   10.0
DESVAR 5          T2      1.0     0.001   10.0
DESVAR 6          T3      1.0     0.001   10.0
$
$...Relate the design variables to analysis model properties
$DVPREL1 ID      TYPE     PID      FID      PMIN     PMAX     CO
$+      DVIDD1 COEF1    DVID2   COEF2    ...
$ 
DVPREL1 1          PROD     201     A
1          1.0
DVPREL1 2          PROD     202     A
2          1.0
DVPREL1 3          PROD     203     A
3          1.0
DVPREL1 4          PSHELL   204     T
4          0.2
DVPREL1 5          PSHELL   205     T
5          0.2

```



```

DVPREL1 6      PSHELL 206      T
          6      0.2
$
$...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
DRESP2 10     STWGHT  20
             DTABLE  FIXWGT
             DRESP1  1
DEQATN 20     STWGT(fixwgt,TOTWGT) = TOTWGT - FIXWGT
DTABLE  FIXWGT  90.
DRESP1 2      F1      FREQ           1
$
$ Identify the constraints
$DCONSTR DCID  RID  LALLOW  UALLOW
DCONSTR 10    2     20.
$
$...Optional override of design optimization parameters:
DOPTPRM IPRINT 2      DESMAX 10      DELP  0.5    P1      1
          P2      15
$
PARAM    POST     -1
ENDDATA

```

The lower-bound constraint on the first mode is 20 Hz. The DRESP1 entry with ID=2 is used to designate the first mode and DCONSTR 10 imposes the lower bound of 20. Hz. Note that the FREQ RTYPE has been used on the DRESP1, allowing the constraint specification in Hz. If the EIGN RTYPE had been used, the constraint would have to be specified on the eigenvalue value. Since there is no upper bound imposed on the response, the upper bound field is left blank on the DCONSTR entry.

The objective to be minimized is the structural weight. The DRESP1 entry with ID=1 selects the WEIGHT response type and, since the ATTA and ATTB fields are left blank, the total structural weight in the Z direction is designated. Because the total weight includes the fixed weight of the lumped masses, a DRESP2 is used to subtract this 90. pound fixed weight from the total weight. This subtraction step is not a requirement for this problem, but it does serve to highlight the structural weight and improves the ability of the optimizer to focus on reducing the weight that it has influence over.

Turning to the optimization results of [Listing 8-4](#), we note that hard convergence has been achieved after six design cycles, requiring a total of seven finite element analyses. We note that a feasible design has been achieved as the maximum constraint value is less than the allowable maximum of GMAX=0.005. The slight constraint violation results in a final first natural frequency of 19.941 Hz. as opposed to the 20.0 Hz. requirement. This is considered more than adequate for engineering purposes, but it should be clear that the GMAX limit could be reduced or the DCONSTR bound could be increased to produce a final design that is closer to or above the 20 Hz. requirement. It is seen that the structural weight has been reduced from the 19.20 pounds of the original design to 6.973 pounds.

Listing 8-4 Design History Output for DSOU2

```

*****
* 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	7
NUMBER OF OPTIMIZATIONS W.R.T. APPROXIMATE MODELS	6

OBJECTIVE AND MAXIMUM CONSTRAINT HISTORY

CYCLE NUMBER	OBJECTIVE FROM APPROXIMATE OPTIMIZATION	OBJECTIVE FROM EXACT ANALYSIS	FRACTIONAL ERROR OF APPROXIMATION	MAXIMUM VALUE OF CONSTRAINT
INITIAL			1.920000E+01	-2.947325E-01
1	1.065987E+01	1.066022E+01	-3.220598E-05	-7.206659E-02
2	7.744514E+00	7.744980E+00	-6.008964E-05	-5.249882E-03
3	6.947861E+00	6.948044E+00	-2.635535E-05	6.191349E-03
4	6.985199E+00	6.985268E+00	-9.829910E-06	2.629376E-03
5	6.972710E+00	6.972694E+00	2.188363E-06	2.970886E-03
6	6.972694E+00	6.972694E+00	0.000000E+00	2.970886E-03

0

DESIGN VARIABLE HISTORY

INTERNAL DV. ID.	EXTERNAL DV. ID.	LABEL	INITIAL	: 1	: 2	: 3	: 4	: 5	:
1	1	A1	1.0000E+00	: 7.6546E-01	: 8.5520E-01	: 8.2450E-01	: 8.8108E-01	: 8.7456E-01	:
2	2	A2	1.0000E+00	: 5.0000E-01	: 3.4241E-01	: 4.3515E-01	: 4.4017E-01	: 4.3576E-01	:
3	3	A3	1.0000E+00	: 5.0000E-01	: 2.5000E-01	: 1.2501E-01	: 9.6605E-02	: 1.0639E-01	:
4	4	T1	1.0000E+00	: 4.9996E-01	: 3.1470E-01	: 2.3823E-01	: 2.2935E-01	: 2.2438E-01	:
5	5	T2	1.0000E+00	: 4.9949E-01	: 2.4975E-01	: 2.2405E-01	: 1.9731E-01	: 1.9497E-01	:
6	6	T3	1.0000E+00	: 4.9987E-01	: 2.4994E-01	: 1.2497E-01	: 1.2079E-01	: 1.2475E-01	:

INTERNAL DV. ID.	EXTERNAL DV. ID.	LABEL	:	6	:	7	:	8	:	9	:	10	:	11	:
1	1	A1	:	8.7456E-01	:										
2	2	A2	:	4.3576E-01	:										
3	3	A3	:	1.0639E-01	:										
4	4	T1	:	2.2438E-01	:										
5	5	T2	:	1.9497E-01	:										
6	6	T3	:	1.2475E-01	:										

*** USER INFORMATION MESSAGE 6464 (DOM12E)

RUN TERMINATED DUE TO HARD CONVERGENCE TO AN OPTIMUM AT CYCLE NUMBER = 6.



Cantilevered Plate

This section uses a simple cantilevered plate example to illustrate two important optimization features:

- Multidisciplinary analysis and design
- Reduced basis vectors

The multidisciplinary aspect of the example refers to the fact that a total of four subcases with three different analysis types (STATICS, MODES and MFREQ) are analyzed and each has their own set of design conditions. Reduced basis formulations were introduced in [Designed Properties](#), as an efficient way to reduce the number of independent design variables. This example shows how the DVPREL1 entries can be used to implement these reduced basis formulations.

Other features, such as constraint screening and optimization parameters are also illustrated and commented upon during the description of the input data file.

Suppose we wish to determine the optimum thickness distribution of the cantilever plate in [Figure 8-18](#) such that the structural mass is minimized. Two separate static loading conditions are imposed: 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 surface of the first row of elements along the length of the cantilever. A normal modes analysis is also performed and a constraint is placed on first natural frequency. Finally, a frequency response analysis is performed that uses a modal solution based on the modes computed in the normal modes analysis. The loading is the same tip loading that is applied in the first static subcase except that it is applied across a frequency range extending from 0.0 to 10. Hz. and at a magnitude that is 10% of the statically applied load.

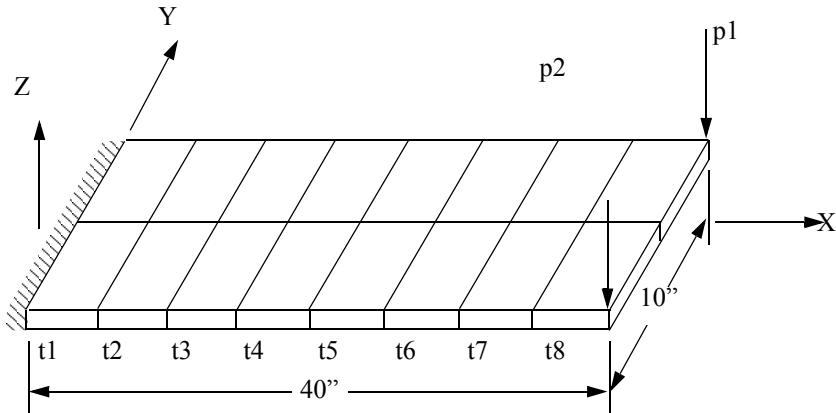


Figure 8-18 Cantilever Plate Thickness Design



Analysis Model Description

2 x 8 array of CQUAD4 elements

Material:

$E = 1.0 \text{E}+7 \text{ psi}$

Poisson ratio = 0.33

Weight density = 0.1 lb/in³

Two static load conditions:

Tip loading:

Two 500 lb loads in -z direction

Pressure loading:

Uniform at 7.5 lb/in²

One normal modes analysis

One modal frequency analysis

Two 50 lb loads in z-direction

Tip loading:

Structural damping of 10%

Design Model Description

Objective:

Structural weight minimization

Design variables:

Basis functions:

Constant, linear and quadratic in x direction to describe plate element thicknesses

Constraints:

ANALYSIS = STATICS

$-2.0 \leq \text{Tip Displacement} \leq 2.0$

von Mises Stress $\leq 29 \text{ksi}$

ANALYSIS = MODES

First natural frequency $\geq 1.5 \text{ Hz}$

ANALYSIS = MFREQ

Magnitude of tip displacement ≤ 2.0

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:

$$\begin{Bmatrix} t_1 \\ t_2 \\ t_3 \\ \vdots \\ t_8 \end{Bmatrix} = \alpha_1 \begin{Bmatrix} 1.0 \\ 1.0 \\ 1.0 \\ \vdots \\ 1.0 \end{Bmatrix} + \alpha_2 \begin{Bmatrix} 1.0 \\ 0.875 \\ 0.75 \\ \vdots \\ 0.125 \end{Bmatrix} + \alpha_3 \begin{Bmatrix} 1.0 \\ 0.7656 \\ 0.5625 \\ \vdots \\ 0.0156 \end{Bmatrix} \quad (8-3)$$



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 . Although DLINK entries could have been used to create these relationships of (you may want to try this exercise on your own), this example applies the relations of [Equation \(8-3\)](#) directly on the DVPREL1 entries.

[Listing 8-5](#) begins with four subcases that define the design task. Subcases 1 and 2 perform static analysis and impose differing load conditions while sharing the imposed design constraints (DESSUB=10). Subcase 3 performs a normal modes analysis with the METHOD=100 identifying the EIGRL Bulk Data entry that specifies the eigenvalue requests while the DESSUB=30 points to the DCONSTR that imposes a limit on the first natural frequency. Subcase 4 performs a modal frequency response analysis (ANALYSIS=MFREQ) and uses the same METHOD=100 specification for the eigenanalysis. For the fourth subcase, the PHASE part of the DISP(PHASE,SORT2)=100 acts to signify that the magnitude/phase representations is to be used not only in the printed and stored results, but also for the FRDISP request on DRESP1 ID=230. The SORT2 request and the 100 have no effect on the design response.

Listing 8-5 Input File for DSOUG3

```
ID MSC  DSOUG3 $ v2004 ehj 25-Jun-2003
TIME 10
SOL 200      $ OPTIMIZATION
CEND
TITLE      = CANTILEVERED PLATE - DSOUG3
SUBTITLE   = REDUCED BASIS FORMULATION - MULTIDISPLINARY OPTIMIZATION
SPC        = 100
DISP       = ALL
STRESS     = ALL
DESOBJ(MIN) = 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
SUBCASE 3
  ANALYSIS = MODES
  LABEL    = NORMAL MODES ANALYSIS
  STRESS   = NONE
  METHOD   = 100
  DESSUB   = 30
SUBCASE 4
  ANALYSIS = MFREQ
  METHOD   = 100
  SET 100  = 9,19,29
  DISP(PHASE,SORT2) = 100
  STRESS   = NONE
  LABEL    = MODAL FREQUENCY ANALYSIS
  LOADSET  = 2000
  DLOAD    = 1000
  DESSUB   = 40
  FREQ     = 1000
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      500.     0.0     0.0     -1.0
FORCE    300     29      500.     0.0     0.0     -1.0

```



```

PLOAD2 310    7.5     1      THRU     8
PLOAD2 310    7.5    11      THRU    18
$
EIGRL   100                               10
$
LSEQ    2000    28     300
RLOAD1 1000    28                               29
TABLED1 29
          0.0     0.1    1000.0    0.1      ENDT
FREQ1  1000    0.0     0.10    100
PARAM   G      0.10
$
$-----
$ DESIGN MODEL:
$-----
$

$...DEFINE THE DESIGN VARIABLES
$DESVAR ID      LABEL    XINIT    XLB      XUB      DELXV
DESVAR  10      ALPH1    1.0
DESVAR  20      ALPH2    1.0
DESVAR  30      ALPH3    1.0
$
$...EXPRESS ANALYSIS MODEL PROPERTIES LINEARLY IN TERMS OF
$   DESIGN VARIABLES
$DVPREL1 ID      TYPE     PID      FID      PMIN      PMAX      CO
$+ DVID1 COEF1  DVID2 COEF2 ...
DVPREL1 1      PSHELL  1       T
          10     1.0 20     1.000    30        1.0
DVPREL1 2      PSHELL  2       T
          10     1.0 20     0.875    30        0.7656
DVPREL1 3      PSHELL  3       T
          10     1.0 20     0.750    30        0.5625
DVPREL1 4      PSHELL  4       T
          10     1.0 20     0.625    30        0.3906
DVPREL1 5      PSHELL  5       T
          10     1.0 20     0.500    30        0.250
DVPREL1 6      PSHELL  6       T
          10     1.0 20     0.375    30        0.1406
DVPREL1 7      PSHELL  7       T
          10     1.0 20     0.250    30        0.0625
DVPREL1 8      PSHELL  8       T
          10     1.0 20     0.125    30        0.0156
$
$...IDENTIFY THE DESIGN RESPONSES
$DRESP1 ID      LABEL    RTYPE    PTYPE    REGION    ATTA    ATTB    ATT1
$+ ATT2 ...
$   STATIC VON MISES STRESS IN SHELL ELEMENTS
DRESP1  2      S12      STRESS   ELEM      9           1
          2       3       4       5       6       7       8
$   STATIC DISPLACEMENT AT THE TIP
DRESP1  33     D1      DISP      3           19
$   FIRST NATURAL FREQUENCY
DRESP1  130     FFREQ    FREQ      1
$   DEFINE THE RESPONSES IN THE DYNAMIC RESPONSE ANALYSIS
DRESP1  230     TDISP    FRDISP    3           19
$
$...DEFINE THE WEIGHT RESPONSE TO BE USED AS THE OBJECTIVE FUNCTION:
DRESP1  35     W      WEIGHT
$
$...DEFINE THE STATIC DESIGN CONSTRAINTS
$DCONSTR DCID    RID      LALLOW   UALLOW
DCONSTR 10     2      -29000. 29000.

```



```

DCONSTR 10      33      -2.      2.
$...DEFINE THE NORMAL MODES DESIGN CONSTRAINT
DCONSTR 30      130      1.5
$...DEFINE THE FREQUENCY RESPONSE DESIGN CONSTRAINTS
DCONSTR 40      230      2.0
$
$...OVERRIDE OPTIMIZATION PARAMETER DEFAULTS (OPTIONAL)
DOPTPRM DESMAX 20      P1   1      P2      15      iprint      7
$
PARAM    NASPRT    1
PARAM    POST      -1
ENDDATA

```

Selected results from the .f06 file from this run are shown in [Listing 8-6](#) and [Listing 8-7](#). The first set of data shows the print of final design information, including the objective, the design variable values, the final property values, and the final responses. As expected, the final design has the thickest elements at the root and the thickness decrease, monotonically along the span of the plate towards the tip.

The final responses show that only one response is close to its limiting value: the displacement of the structure under the first static load. This single constraint condition is therefore dictating the design of the entire structure to produce the thickness distribution shown above.

Listing 8-6 Final Results for DSOUG3

```

*****
*                                              DESIGN OPTIMIZATION
*
*                                              FINAL ANALYSIS
*
***** ANALYSIS RESULTS BASED ON THE FINAL DESIGN *****
----- DESIGN OBJECTIVE -----


| INTERNAL<br>RESPONSE<br>ID | TYPE<br>OF<br>RESPONSE | LABEL  | MINIMIZE<br>OR<br>MAXIMIZE | SUPERELEMENT<br>ID | SUBCASE<br>ID | VALUE      |
|----------------------------|------------------------|--------|----------------------------|--------------------|---------------|------------|
| 1                          | DRESP1                 | WEIGHT | MINIMIZE                   | 0                  | 0             | 3.7018E+01 |


----- DESIGN VARIABLES -----


| INTERNAL<br>ID | DESVAR<br>ID | LABEL | LOWER<br>BOUND | VALUE      | UPPER<br>BOUND |
|----------------|--------------|-------|----------------|------------|----------------|
| 1              | 10           | ALPH1 | -1.0000E+20    | 3.6149E-01 | 1.0000E+20     |
| 2              | 20           | ALPH2 | -1.0000E+20    | 8.4961E-01 | 1.0000E+20     |
| 3              | 30           | ALPH3 | -1.0000E+20    | 2.1597E-01 | 1.0000E+20     |


----- DESIGNED PROPERTIES -----


| PROPERTY<br>TYPE | PROPERTY<br>ID | PROPERTY<br>NAME | TYPE OF<br>PROPERTY | LOWER<br>BOUND | VALUE      | UPPER<br>BOUND |
|------------------|----------------|------------------|---------------------|----------------|------------|----------------|
| PSHELL           | 1              | T                | DVPREL1             | 1.0000E-15     | 1.4271E+00 | 1.0000E+20     |
| PSHELL           | 2              | T                | DVPREL1             | 1.0000E-15     | 1.2702E+00 | 1.0000E+20     |
| PSHELL           | 3              | T                | DVPREL1             | 1.0000E-15     | 1.1202E+00 | 1.0000E+20     |
| PSHELL           | 4              | T                | DVPREL1             | 1.0000E-15     | 9.7685E-01 | 1.0000E+20     |
| PSHELL           | 5              | T                | DVPREL1             | 1.0000E-15     | 8.4028E-01 | 1.0000E+20     |
| PSHELL           | 6              | T                | DVPREL1             | 1.0000E-15     | 7.1046E-01 | 1.0000E+20     |
| PSHELL           | 7              | T                | DVPREL1             | 1.0000E-15     | 5.8739E-01 | 1.0000E+20     |
| PSHELL           | 8              | T                | DVPREL1             | 1.0000E-15     | 4.7106E-01 | 1.0000E+20     |


```



RESPONSES IN DESIGN MODEL																
(N/A - BOUND NOT ACTIVE OR AVAILABLE)																
----- WEIGHT RESPONSE -----																
INTERNAL ID	DRESP1 ID	RESPONSE LABEL	ROW ID	COLUMN ID	LOWER BOUND	VALUE		UPPER BOUND								
1	35	W	3	3	N/A	3.7018E+01		N/A								
FINAL ANALYSIS SUBCASE = 1																
----- DISPLACEMENT RESPONSES -----																
INTERNAL ID	DRESP1 ID	RESPONSE LABEL	GRID ID	COMPONENT NO.	LOWER BOUND	VALUE		UPPER BOUND								
2	33	D1	19	3	-2.0000E+00	-2.0053E+00		N/A								
----- STRESS RESPONSES -----																
INTERNAL ID	DRESP1 ID	RESPONSE LABEL	ELEMENT ID	VIEW ELM ID	COMPONENT NO.	LOWER BOUND	VALUE	UPPER BOUND								
3	2	S12	5		9	N/A	1.4778E+04	2.9000E+04								
4	2	S12	6		9	N/A	1.4697E+04	2.9000E+04								
FINAL ANALYSIS SUBCASE = 2																
----- DISPLACEMENT RESPONSES -----																
INTERNAL ID	DRESP1 ID	RESPONSE LABEL	GRID ID	COMPONENT NO.	LOWER BOUND	VALUE		UPPER BOUND								
5	33	D1	19	3	N/A	1.7815E+00		2.0000E+00								
0	SUBCASE 1															
FINAL ANALYSIS SUBCASE = 2																
----- STRESS RESPONSES -----																
INTERNAL ID	DRESP1 ID	RESPONSE LABEL	ELEMENT ID	VIEW ELM ID	COMPONENT NO.	LOWER BOUND	VALUE	UPPER BOUND								
6	2	S12	2		9	N/A	1.4565E+04	2.9000E+04								
FINAL ANALYSIS SUBCASE = 3																
----- NORMAL MODE RESPONSES -----																
INTERNAL ID	DRESP1 ID	RESPONSE LABEL		MODE NO.	LOWER BOUND	VALUE		UPPER BOUND								
7	130	FFREQ		1	1.5000E+00	1.7120E+00		N/A								
FINAL ANALYSIS SUBCASE = 4																
----- FREQUENCY DISPLACEMENT RESPONSES -----																
INTERNAL ID	DRESP1 ID	RESPONSE LABEL	GRID ID	COMPONENT NO.	FREQUENCY	LOWER BOUND	VALUE	UPPER BOUND								
8	230	TDISP	19	3	1.6000E+00	N/A	1.1551E+00	2.0000E+00								
9	230	TDISP	19	3	1.7000E+00	N/A	1.8253E+00	2.0000E+00								
10	230	TDISP	19	3	1.8000E+00	N/A	1.2531E+00	2.0000E+00								

The summary of design cycle history in Listing 8-7 indicates that the initial design is well within the imposed design requirements. This is indicated by the N/A that is shown in the “MAXIMUM VALUE OF CONSTRAINT” column which indicates that none of the constraints exceeded the -0.5 value that is the default for designating a constraint as being active. After six iterations, a minimum weight design has been achieved that satisfies all constraints and has reduced the structural weight from 78.375 to 37.018.



Listing 8-7 Design History for DSOUG3

```
*****
          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           8  

NUMBER OF OPTIMIZATIONS W.R.T. APPROXIMATE MODELS      7  

OBJECTIVE AND MAXIMUM CONSTRAINT HISTORY  

-----  


| CYCLE<br>NUMBER | OBJECTIVE FROM              |                   | FRACTIONAL ERROR    |                  | MAXIMUM VALUE |  |
|-----------------|-----------------------------|-------------------|---------------------|------------------|---------------|--|
|                 | APPROXIMATE<br>OPTIMIZATION | EXACT<br>ANALYSIS | OF<br>APPROXIMATION | OF<br>CONSTRAINT |               |  |
| INITIAL         | 7.843750E+01                |                   |                     |                  | N/A           |  |
| 1               | 6.283712E+01                | 6.283225E+01      | 7.752974E-05        |                  | N/A           |  |
| 2               | 5.031821E+01                | 5.031795E+01      | 5.231018E-06        |                  | -4.653676E-01 |  |
| 3               | 4.143894E+01                | 4.144248E+01      | -8.551258E-05       |                  | -2.308573E-01 |  |
| 4               | 3.787624E+01                | 3.787529E+01      | 2.507861E-05        |                  | -2.846581E-02 |  |
| 5               | 3.745531E+01                | 3.745506E+01      | 6.518228E-06        |                  | -6.227493E-04 |  |
| 6               | 3.739384E+01                | 3.739399E+01      | -3.978533E-06       |                  | 5.829334E-04  |  |
| 7               | 3.736565E+01                | 3.736588E+01      | -6.023333E-06       |                  | 3.526211E-04  |  |

1 CANTILEVERED PLATE - DSOUG3                         SEPTEMBER 7, 2009 MSC NASTRAN 9/ 4/09  

PAGE 483  

REDUCED BASIS FORMULATION - MULTIDISPLINARY OPTIMIZATION  

0 MODAL FREQUENCY ANALYSIS                            SUBCASE 4  

DESIGN VARIABLE HISTORY  

-----  


| INTERNAL   EXTERNAL                                                                            |                                       |  |  |  |  |  |  |  |  |  |
|------------------------------------------------------------------------------------------------|---------------------------------------|--|--|--|--|--|--|--|--|--|
| DV. ID.   DV. ID.                                                                              | LABEL   INITIAL : 1 : 2 : 3 : 4 : 5 : |  |  |  |  |  |  |  |  |  |
| 1   10   ALPH1   1.0000E+00 : 7.9764E-01 : 6.4107E-01 : 5.0823E-01 : 3.9048E-01 : 3.9783E-01 : |                                       |  |  |  |  |  |  |  |  |  |
| 2   20   ALPH2   1.0000E+00 : 8.2125E-01 : 6.2892E-01 : 5.3005E-01 : 5.4091E-01 : 5.3530E-01 : |                                       |  |  |  |  |  |  |  |  |  |
| 3   30   ALPH3   1.0000E+00 : 7.8111E-01 : 6.6038E-01 : 5.7647E-01 : 6.3285E-01 : 5.9593E-01 : |                                       |  |  |  |  |  |  |  |  |  |


| INTERNAL   EXTERNAL                        |                                   |  |  |  |  |  |  |  |  |  |  |
|--------------------------------------------|-----------------------------------|--|--|--|--|--|--|--|--|--|--|
| DV. ID.   DV. ID.                          | LABEL   6 : 7 : 8 : 9 : 10 : 11 : |  |  |  |  |  |  |  |  |  |  |
| 1   10   ALPH1   4.1056E-01 : 4.1485E-01 : |                                   |  |  |  |  |  |  |  |  |  |  |
| 2   20   ALPH2   5.2782E-01 : 5.3592E-01 : |                                   |  |  |  |  |  |  |  |  |  |  |
| 3   30   ALPH3   5.7074E-01 : 5.4676E-01 : |                                   |  |  |  |  |  |  |  |  |  |  |

*** USER INFORMATION MESSAGE 6464 (DOM12E)  

RUN TERMINATED DUE TO HARD CONVERGENCE TO AN OPTIMUM AT CYCLE NUMBER = 7.
```

Figure 8-19 depicts results from the two static subcases for the final design that have been produced using Patran. This figure is representative of the many results can be produced from results in the .op2 database that was generated based on the PARAM POST -1 in the input data file of Listing 8-5. Other items in the input data file that affect the amount of data written to the output database are the case control commands, such as DISP=ALL and STRESS=ALL and the PARAM NASPRT 1 that is used to output data recovery results at every design cycle.



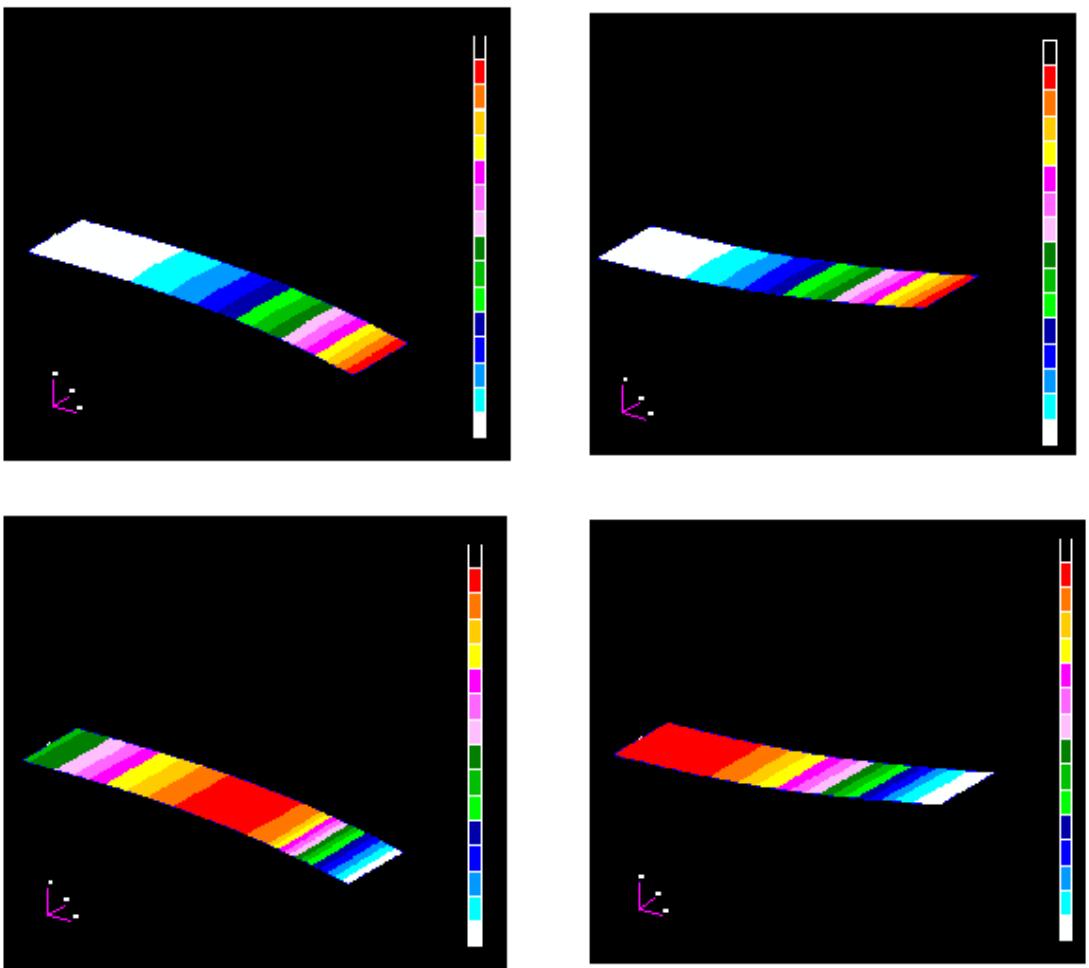


Figure 8-19 Representative Displacement and Stress Results for the Final Design of the Cantilevered Beam



Stiffened Plate

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 8-20](#) and includes a plate with a hat stiffener. 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 the thickness of the hat stiffener are allowed to vary. The boundary condition creates a simply supported condition with the plate also restrained in the x direction along $x=0.0$. 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 hat. The example illustrates how the beam library can be utilized to simplify the modeling and design tasks and how the beam offset relations can be adjusted as the structural properties change.

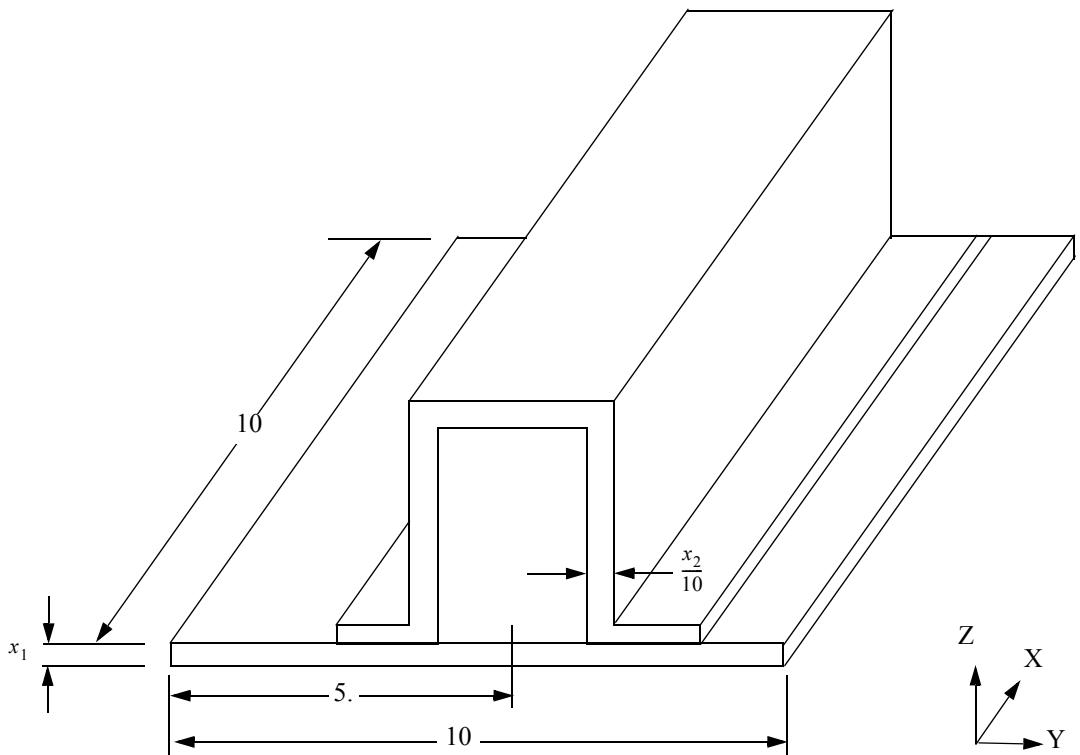


Figure 8-20 Plate with a HAT Stiffener

Analysis Model Description

The input file for this example is shown in Listing 8-8.

16 CQUAD4 elements to model uniform thickness base plate

4 CBAR elements to model the hat stiffener. The HAT type is selected on the PBARL entry.

Material: E = 1.0E7 psi

Poisson ratio = 0.33

Weight density = 0.283 lbs/in³

Listing 8-8 Input File for DSQUG4

```

$ field entries left justified for clarity
$ cbar entry line 1: vector v is z direction
$ cbar entry line 2: w3a and w3b that is fields 6 and 9
$ rename t-box as hatdim2
$
ID MSC DSOUG4 $ v2004 ehj 25-Jun-2003
TIME 10
SOL 200      $ OPTIMIZATION
CEND
$
TITLE = STATIC ANALYSIS OF A STIFFENED PLATE          DSOUG4
ECHO = BOTH
DISP = ALL
STRESS = ALL
SPC = 1
ANALYSIS = STATICS
DSAPRT(START=1,END=LAST)=ALL
DESOBJ(MIN) = 15      $ OBJECTIVE FUNCTION DEFINITION
$                               (MIN IS THE DEFAULT)
SUBCASE 1
  LABEL = LOAD CONDITION 1
  LOAD = 1
  DESSUB = 100      $ CONSTRAINT DEFINITION
SUBCASE 2
  LABEL = LOAD CONDITION 2
  LOAD = 2
  DESSUB = 200      $ CONSTRAINT DEFINITION
BEGIN BULK
param, post, -1
PARAM, NASPRT, 1
$
$-----
$ 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	
\$						
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
\$						
\$11111112222222333333334444444455555555666666667777777888888899999999						
CBAR	31	3	10200	10201	0.0	0.0
					1.0	1.575
CBAR	32	3	10201	10202	0.0	0.0
					1.0	1.575
CBAR	33	3	10202	10203	0.0	0.0
					1.0	1.575
CBAR	34	3	10203	10204	0.0	0.0
					1.0	1.575
						1.575
\$						
PSHELL	1	1	0.15	1		
PSHELL	2	1	0.2	1		
\$11111112222222333333334444444455555555666666667777777888888899999999						
PBARL	3	1		hat		
	3.0	0.1	2.0	0.9		
\$						
MAT1	1		1.0E+7		0.33	0.283
\$						
FORCE	1		10004		2000.0	1.0
FORCE	1		10104		2000.0	1.0
FORCE	1		10204		2000.0	1.0
FORCE	1		10304		2000.0	1.0
FORCE	1		10404		2000.0	1.0
FORCE	2		10203		10000.0	0.0
PLOAD2	1		50.	1	THRU	16
\$						
SPC1	1		1236	10000		
SPC1	1		136	10100	10300	10400
SPC1	1		36	10001	10002	10003
SPC1	1		36	10401	10402	10403
SPC1	1		3	10204	10404	10304



```

SPC1    1      1.3      10200
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        HATDIM2  1.0     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       T        0.01
+DP1   1        1.0
$  

$...Express BOX THICKNESS as a function of x2:  

DVPREL1 3        PBARL    3       DIM2
      2        0.1
$.   EXPRESS BOX OFFSET LOCATIONS AS A FUNCTION OF PLATE THICKNESS AND
$   FIXED BOX DIMENSIONS
$VCREL1 ID      TYPE     EID      CPNAME    CPMIN    CPMAX      C0          +
$+    DVIDD1 COEF1   DVID2   COEF2    ...
DVCREL1 10       CBAR    31      W3A
      1        0.5
DVCREL1 20       CBAR    32      W3A
      1        0.5
DVCREL1 30       CBAR    33      W3A
      1        0.5
DVCREL1 40       CBAR    34      W3A
      1        0.5
DVCREL1 50       CBAR    31      W3B
      1        0.5
DVCREL1 60       CBAR    32      W3B
      1        0.5
DVCREL1 70       CBAR    33      W3B
      1        0.5
DVCREL1 80       CBAR    34      W3B
      1        0.5
$...Identify the design responses:  

$  

$DRESP1 ID      LABEL     RTYPE    PTYPE     REGION    ATTA     ATTB     ATT1      +
$+    ATT2    ...
DRESP1 1        SBARA   STRESS   PBAR
DRESP1 2        SBARB   STRESS   PBAR
DRESP1 3        S13    STRESS   PSHELL
DRESP1 6        S16    STRESS   PSHELL
DRESP1 13       D1      DISP
DRESP1 14       D2      DISP
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   20      13     -0.1      0.1
DCONSTR   30      14     -0.03     0.03
$
$DCONADD DCID    DC1      DC2      ...
$ summed constraint set for subcase 1
DCONADD  100     10      20
$ summed constraint set for subcase 2
DCONADD  200     10      30
$
$...Optional override of optimization parameters:
$.
DOPTPRM IPRINT  7      DESMAX  20      DELP    0.5      P1      1      +
+          P2      15
$ (DELP=0.5 allows larger moves thus overcoming constraint
$ violations quicker)
$.
ENDDATA

```

Design Model Description

Objective: Structural weight minimization

Design variables: Plate thickness and hat stiffener thickness

Constraints:

Strength: von Mises stress $\leq 25,000$ psi at the centroid of plate elements
Maximum stress $\leq 25,000$ psi at both ends of rod elements.

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 express the analysis model properties in terms of design variables:

$$\begin{aligned} t_{\text{plate}} &= 1.0 X_1 \\ t_{\text{hat}} &= 0.1 X_2 \\ w3 &= 1.5 + 0.5 X_1 \end{aligned} \tag{8-4}$$

The third relationship adjusts the beam offset of the for the hat section to be one-half the height of the hat stiffener plus one half the design plate thickness.

In order for a structural response to be used either as an objective or a constraint, it first must be identified on a DRESP1 Bulk Data entry. The DRESP1 entries 1 and 2, for example, identify the maximum stress at ends A and B of a bar element. (Item 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 Item 14 is the maximum stress at end B. (See [Item Codes](#) of the *MSC 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 and 6 identify the von Mises stresses at Z1 and Z2 for half the elements in the the plate (only one side of the plate needs the responses in this case due to the symmetry of the structure and the loading about line $y=5.0$. Separate DRESP1 entries identify displacement responses at grid points. These responses are constrained by the bounds set using a corresponding set of DCONSTR entries. The DCONADD entry is used to accumulate the design constraints that are appropriate for each subcase. Note that the stress constraints are used in both subcases while a distinct displacement constraint is applied for the two loading conditions.

Extensive output for this example is presented in [Design Optimization Output](#). The SUMMARY OF DESIGN CYCLE HISTORY is repeated here [Listing 8-9](#) and it is seen that the optimization task required 3 design cycles with 4 finite element analyses. Note that it was possible to both overcome design constraints while reducing the weight objective function. The MAXIMUM VALUE OF CONSTRAINT column shows the original design had a response that has a margin of error over 16% while the final design has a maximum violation that is much less than the 0.5% allowed by the GMAX parameter. The weight decreases from 6.962 to 5.484.

Listing 8-9 Design History Output for DSOUG4

```
*****
SUMMARY OF DESIGN CYCLE HISTORY
*****
(HARD CONVERGENCE ACHIEVED)
(SOFT CONVERGENCE ACHIEVED)
NUMBER OF FINITE ELEMENT ANALYSES COMPLETED      4
NUMBER OF OPTIMIZATIONS W.R.T. APPROXIMATE MODELS   3
OBJECTIVE AND MAXIMUM CONSTRAINT HISTORY
-----
          OBJECTIVE FROM          OBJECTIVE FROM          FRACTIONAL ERROR          MAXIMUM VALUE
CYCLE      APPROXIMATE      EXACT                  OF                      OF
NUMBER     OPTIMIZATION    ANALYSIS               APPROXIMATION           CONSTRAINT
-----
INITIAL          6.961800E+00                         -1.639155E-01
1            5.369087E+00          5.369616E+00        -9.839355E-05       7.175805E-02
2            5.484202E+00          5.484169E+00        5.999407E-06       1.318906E-03
3            5.484169E+00          5.484169E+00        0.000000E+00       1.318906E-03
-----
1      STATIC ANALYSIS OF A STIFFENED PLATE          DSOUG4        SEPTEMBER 7, 2009 MSC NASTRAN 9/ 4/09
PAGE    114
0
DESIGN VARIABLE HISTORY
-----
INTERNAL| EXTERNAL |
DV. ID. | DV. ID. | LABEL | INITIAL : 1 : 2 : 3 : 4 : 5
-----
1 | 1 | T-PLATE | 1.5000E-01 : 1.0852E-01 : 1.1265E-01 : 1.1265E-01 :
2 | 2 | HATDIM2 | 1.0000E+00 : 8.4324E-01 : 8.4238E-01 : 8.4238E-01 :
*** USER INFORMATION MESSAGE 6464 (DOM12E)
RUN TERMINATED DUE TO HARD CONVERGENCE TO AN OPTIMUM AT CYCLE NUMBER =      3.
```



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 8-21](#). 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/in² 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.

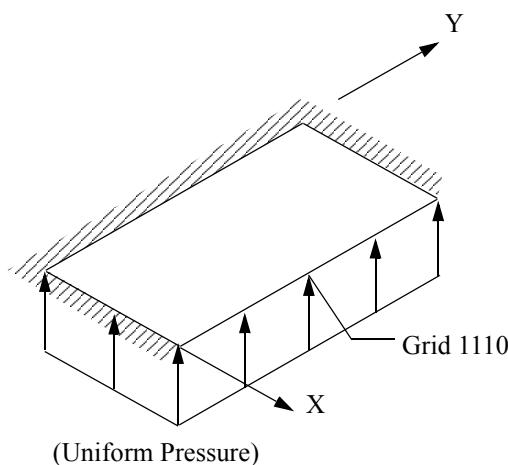


Figure 8-21 Pressure-Loaded Flat Plate

[Figure 8-22](#) 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 which controls the single element connected to GRID 1110.



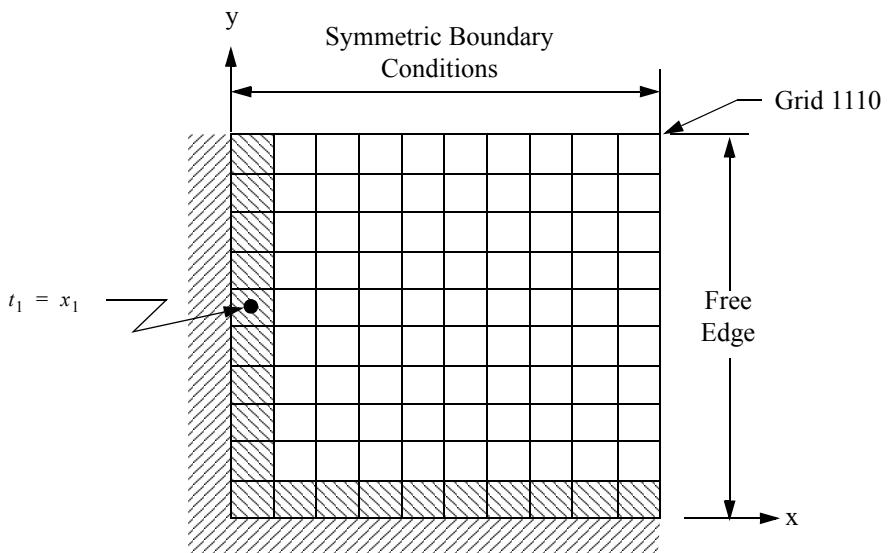


Figure 8-22 Clamped-Free Plate

The dynamic excitation is applied from 20. to 200. Hz. The objective is to minimize the root mean square (RMS) of transverse displacements at GRID 1110. MSC Nastran provides several alternative ways of computing RMS quantities and three of these are illustrated here. In the first variation, the SQRT and SSQ arguments in the DEQATN functions and used to minimize an objective of the form:

$$\min \phi = \sqrt{\sum_{i=20}^{100} (u_{z, 1110}^i)^2 + 2 \sum_{i=51}^{100} (u_{z, 1110}^{2i})^2} \quad (8-5)$$

The above notation indicates 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. The square root of this quantity, therefore, provides a close approximation to the rms response over the frequency range. This particular form was selected simply for illustrative purposes and to highlight the versatility of the DEQATN feature.

The second variation uses the RSS function directly on the DRESP1 entry. In this method, a root sum of squares is performed for all the excitation frequencies in the analysis. When this response is chosen as the objective, the resulting objective function is of the form:

$$\min \phi = \sqrt{\sum_{i=20}^{200} (u_{z, 1110}^i)^2} \quad (8-6)$$

The third variation selects the RMSDISP response type on the DRESP1 entry. The input power spectral input is specified to be a unit value over the excitation range and the excitations frequencies are select so that the response value computed from the RMSDISP response is virtually identical to that of Equation (8-6).



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. 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 of modes increased, if necessary.

The input file [Listing 8-10](#) shows the portion of the input that is common for all three versions of the example. 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:

$$7.99 \leq vol \leq 8.01 \quad (8-7)$$

Listing 8-10 Common Portion of Input File DSOUG7

```

ID MSC  DSOUG7A $
TIME 200
SOL 200
CEND
TITLE = Synthesis of Responses across Different Frequencies: DSOUG7A
SET 10 = 1110
DISPL(PHASE,SORT1) = 10    $ MAGNITUDE/PHASE REPRESENTATION FOR RESPONSE
$                                ANALYSIS AS WELL AS SENSITIVITY ANALYSIS
desglb = 10
subcase 1
SPC          = 100
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
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)  *
=8
CQUAD4  102      1      101     102     202     201
=      *(1)     =      *(1)     * (1)    * (1)    *
=7
PSHELL  1      150      .08      150
$ . . . ELEMENT GROUP 2:
CQUAD4  202      2      201     202     302     301
=      *(100)   =      *(100)   * (100)  * (100)  *
=7
CQUAD4  203      2      202     203     303     302
=      *(1)     =      *(1)     * (1)    * (1)    *
=6
PSHELL  2      150      .08      150
$ . . . ELEMENT GROUP 3:
CQUAD4  303      3      302     303     403     402
=      *(100)   =      *(100)   * (100)  * (100)  *
=6
CQUAD4  304      3      303     304     404     403
=      *(1)     =      *(1)     * (1)    * (1)    *
=5
PSHELL  3      150      .08      150
$ . . . ELEMENT GROUP 4:
CQUAD4  404      4      403     404     504     503
=      *(100)   =      *(100)   * (100)  * (100)  *
=5
CQUAD4  405      4      404     405     505     504
=      *(1)     =      *(1)     * (1)    * (1)    *
=4

```



```

PSHELL 4      150      .08      150
$...ELEMENT GROUP 5:
CQUAD4 505    5        504      505      605      604
=       *(100)   =      * (100)  * (100)  * (100)
=4
CQUAD4 506    5        505      506      606      605
=       *(1)     =      * (1)    * (1)    * (1),
=3
PSHELL 5      150      .08      150
$...ELEMENT GROUP 6:
CQUAD4 606    6        605      606      706      705
=       *(100)   =      * (100)  * (100)  * (100)
=3
CQUAD4 607    6        606      607      707      706
=       *(1)     =      * (1)    * (1)    * (1),
=2
PSHELL 6      150      .08      150
$...ELEMENT GROUP 7:
CQUAD4 707    7        706      707      807      806
=       *(100)   =      * (100)  * (100)  * (100)
=2
CQUAD4 708    7        707      708      808      807
=       *(1)     =      * (1)    * (1)    * (1),
=1
PSHELL 7      150      .08      150
$...ELEMENT GROUP 8:
CQUAD4 808    8        807      808      908      907
=       *(100)   =      * (100)  * (100)  * (100)
=1
CQUAD4 809    8        808      809      909      908
=       *(1)     =      * (1)    * (1)    * (1),
PSHELL 8      150      .08      150
$...ELEMENT GROUP 9:
CQUAD4 909    9        908      909      1009     1008
=       *(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
$
$...EIGRLE EXTRATION INFORMATION - 15 RETAINED MODES
$ EIGRL 500          15      0
$...FREQUENCY DEPENDENT LOADING DATA: (OSCILLATORY PRESSURE LOAD)
$ RLOAD1 700    730      800
$ 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
$ 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 T .01
1 1.0
DVPREL1 102 PSHELL 2 T .01
2 1.0
DVPREL1 103 PSHELL 3 T .01
3 1.0
DVPREL1 104 PSHELL 4 T .01
4 1.0
DVPREL1 105 PSHELL 5 T .01
5 1.0
DVPREL1 106 PSHELL 6 T .01
6 1.0
DVPREL1 107 PSHELL 7 T .01
7 1.0
DVPREL1 108 PSHELL 8 T .01
8 1.0
DVPREL1 109 PSHELL 9 T .01
9 1.0
DVPREL1 110 PSHELL 10 T .01
10 1.0
$
DRESP1 201 VOLUME VOLUME
include 'dsoug7_deq.inc'
$include 'dsoug7_rss.inc'
$include 'dsoug7_rms.inc'
$
$
DCONSTR 10 201 7.99 8.01
doptprm desmax 40 p1 1 p2 8 convl 0.01
$.....2.....3.....4.....5.....6.....7.....8.....9.....0
$
PARAM POST - 1
ENDDATA

```

The three versions of the objective are developed by including one of three files. The first is named dsoug7_deq.inc and is given in [Listing 8-11](#). The RMS response is 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, a capability that is considered obsolete in the presence of preprocessors). 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. The DEQATN uses the SSQ and SQRT functions to create the objective given by [Equation \(8-5\)](#).

Listing 8-11 Include File dsoug7-deq.inc

FREQ1	740	20.	1.	179				
DRESP1	20	g1110L	FRDISP					
=	* (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)
=sqrt(ssq(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) + 2.0 *
ssq( 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)
    
```

The input for the second version is given in [Listing 8-12](#) and it is seen that the RSS specification of the ATTB field of the DRESP1 with ID=1 is sufficient to generate a root mean square response. In this case, all 181 frequencies contribute to the response.

Listing 8-12 Include File dsoug7-rss.inc



\$
FREQ1 740 20. 1. 180
DRESP1 1 q1110L FRDISP 3 rss 1110

The unique input for the third version is given in Listing 8-13. The excitation frequencies have been changed so that 2 Hz intervals are taken at excitation frequencies above 100. Hz. In this case, a DRESP1 with an RMSDISP response type is used. The ATTB field of the DRESP1 points to the RANDPS entry with ID=20. This random input power spectrum has a value of 1.0 across the range of excitation frequencies, with the net result that the RMSDISP is virtually identical to the DEQATN of the first variation given by Equation (8-5).

Listing 8-13 Include File dsoug7-rms.inc

```

$ FREQ1    740      20.     1.      79
$ FREQ1    740      102.    2.      49
$ DRESP1   1          G1110L RMSDISP
$ RANDPS   20         1        1       1.0           3          20      1110
$ TABRND1  20         0.0     1.0     1000.0      ENDT

```

Printed output results for the three options are given in Listing 8-14, Listing 8-15, and Listing 8-16.

Figure 8-23 plots the final thickness distributions for the three variations. The results are seen to be very similar for the different methods. The RMS value decreases from 15.20 to around 12.0. All methods converged in the five design cycles.

Listing 8-14 Design History Results Using the DEQATN Option

OBJECTIVE FROM APPROXIMATE OPTIMIZATION		OBJECTIVE FROM EXACT ANALYSIS		FRACTIONAL ERROR OF APPROXIMATION	MAXIMUM VALUE OF CONSTRAINT
CYCLE NUMBER					
INITIAL		1.519667E+01			-1.249123E-03
1	1.443932E+01	1.365850E+01	5.716709E-02		-1.247873E-03
2	1.324025E+01	1.271585E+01	4.124225E-02		-1.214774E-03
3	1.252116E+01	1.248591E+01	2.822777E-03		-1.217750E-03
4	1.232194E+01	1.216405E+01	1.298024E-02		-1.230133E-03
5	1.195737E+01	1.212859E+01	-1.411712E-02		-1.233466E-03
1 SYNTHESIS OF RESPONSES ACROSS DIFFERENT FREQUENCIES: DSOUG7A				SEPTEMBER 19, 2001	MSC.NASTRAN 4 / 9/01
PAGE 87					
0					SUBCASE 1
DESIGN VARIABLE HISTORY					
INTERNAL DV. ID.	EXTERNAL DV. ID.	LABEL	INITIAL	: 1 : 2 : 3 : 4 : 5 : :	
1	1	T1	8.00000E-02	: 9.60000E-02 : 9.3140E-02 : 1.0635E-01 : 8.9943E-02 : 1.0793E-01 :	
2	2	T2	8.00000E-02	: 7.9898E-02 : 8.9400E-02 : 8.3661E-02 : 9.3010E-02 : 7.7354E-02 :	
3	3	T3	8.00000E-02	: 6.4675E-02 : 7.2220E-02 : 7.0797E-02 : 7.1599E-02 : 6.7977E-02 :	
4	4	T4	8.00000E-02	: 6.5901E-02 : 5.2721E-02 : 4.4020E-02 : 4.4555E-02 : 4.2334E-02 :	
5	5	T5	8.00000E-02	: 7.3891E-02 : 6.0025E-02 : 5.1219E-02 : 4.7529E-02 : 4.7742E-02 :	
6	6	T6	8.00000E-02	: 7.9610E-02 : 7.1892E-02 : 7.0936E-02 : 7.5427E-02 : 7.3412E-02 :	
7	7	T7	8.00000E-02	: 8.5574E-02 : 8.6322E-02 : 8.9026E-02 : 9.5125E-02 : 9.4058E-02 :	
8	8	T8	8.00000E-02	: 9.6000E-02 : 1.1396E-01 : 1.2035E-01 : 1.2918E-01 : 1.3052E-01 :	
9	9	T9	8.00000E-02	: 9.6000E-02 : 1.1520E-01 : 1.2517E-01 : 1.3843E-01 : 1.4504E-01 :	
10	10	T10	8.00000E-02	: 9.6000E-02 : 1.1520E-01 : 1.2082E-01 : 1.2812E-01 : 1.3241E-01 :	
*** USER INFORMATION MESSAGE 6464 (DOM12E)					
RUN TERMINATED DUE TO HARD CONVERGENCE TO AN OPTIMUM AT CYCLE NUMBER = 5.					



Listing 8-15 Design History Results for the RSS Option on DRESP1

```

OBJECTIVE AND MAXIMUM CONSTRAINT HISTORY
-----
CYCLE      OBJECTIVE FROM          OBJECTIVE FROM          FRACTIONAL ERROR      MAXIMUM VALUE
NUMBER     APPROXIMATE           EXACT                OF                   OF
          OPTIMIZATION          ANALYSIS             APPROXIMATION        CONSTRAINT
-----
INITIAL    1.444129E+01          1.519907E+01          -1.249123E-03
1          1.366040E+01          5.716482E-02          -1.247873E-03
2          1.324232E+01          1.271818E+01          -1.245088E-03
3          1.252379E+01          1.248799E+01          -1.240851E-03
4          1.232415E+01          1.216508E+01          -1.236614E-03
5          1.195864E+01          1.212917E+01          -1.237628E-03
1          SYNTHESIS OF RESPONSES ACROSS DIFFERENT FREQUENCIES: DSOUG7   SEPTEMBER 19, 2001 MSC.NASTRAN 4 / 9/01
PAGE      450
0          SUBCASE 1
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 : 9.3151E-02 : 1.0633E-01 : 8.9940E-02 : 1.0793E-01 :
2       | 2      | T2   | 8.0000E-02 : 7.9913E-02 : 8.9366E-02 : 8.3660E-02 : 9.2985E-02 : 7.7328E-02 :
3       | 3      | T3   | 8.0000E-02 : 6.4681E-02 : 7.2203E-02 : 7.0783E-02 : 7.1560E-02 : 6.7921E-02 :
4       | 4      | T4   | 8.0000E-02 : 6.5895E-02 : 5.2711E-02 : 4.3998E-02 : 4.4495E-02 : 4.2223E-02 :
5       | 5      | T5   | 8.0000E-02 : 7.3882E-02 : 6.0032E-02 : 5.1232E-02 : 4.7626E-02 : 4.7890E-02 :
6       | 6      | T6   | 8.0000E-02 : 7.9598E-02 : 7.1914E-02 : 7.0961E-02 : 7.5457E-02 : 7.3469E-02 :
7       | 7      | T7   | 8.0000E-02 : 8.5362E-02 : 8.6328E-02 : 9.0353E-02 : 9.5133E-02 : 9.4091E-02 :
8       | 8      | T8   | 8.0000E-02 : 9.6000E-02 : 1.1396E-01 : 1.2036E-01 : 1.2920E-01 : 1.3054E-01 :
9       | 9      | T9   | 8.0000E-02 : 9.6000E-02 : 1.1520E-01 : 1.2518E-01 : 1.3846E-01 : 1.4506E-01 :
10      | 10     | T10  | 8.0000E-02 : 9.6000E-02 : 1.1520E-01 : 1.2083E-01 : 1.2814E-01 : 1.3242E-01 :
*** USER INFORMATION MESSAGE 6464 (DOM12E)
RUN TERMINATED DUE TO HARD CONVERGENCE TO AN OPTIMUM AT CYCLE NUMBER = 5.

```

Listing 8-16 Design History for the RMS Option

```

0          SUBCASE 1
*****
SUMMARY OF DESIGN CYCLE HISTORY
*****
(HARD CONVERGENCE ACHIEVED)

NUMBER OF FINITE ELEMENT ANALYSES COMPLETED      6
NUMBER OF OPTIMIZATIONS W.R.T. APPROXIMATE MODELS  5

OBJECTIVE AND MAXIMUM CONSTRAINT HISTORY
-----
CYCLE      OBJECTIVE FROM          OBJECTIVE FROM          FRACTIONAL ERROR      MAXIMUM VALUE
NUMBER     APPROXIMATE           EXACT                OF                   OF
          OPTIMIZATION          ANALYSIS             APPROXIMATION        CONSTRAINT
-----
INITIAL    1.442682E+01          1.518897E+01          -1.249123E-03
1          1.368694E+01          5.405684E-02          -1.248587E-03
2          1.326691E+01          1.269624E+01          2.155018E-04
3          1.238254E+01          1.212222E+01          2.147395E-02          2.113242E-04
4          1.198582E+01          1.198919E+01          -2.805536E-04          2.107274E-04
5          1.184377E+01          1.188050E+01          -3.091925E-03          2.132339E-04
1          SYNTHESIS OF RESPONSES ACROSS DIFFERENT FREQUENCIES: DSOUG7   SEPTEMBER 19, 2001 MSC.NASTRAN 4 / 9/01
PAGE      56
0          SUBCASE 1
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 : 9.4803E-02 : 1.0653E-01 : 9.1509E-02 : 1.0526E-01 :
2       | 2      | T2   | 8.0000E-02 : 8.1651E-02 : 8.6642E-02 : 8.0803E-02 : 9.0622E-02 : 7.9363E-02 :
3       | 3      | T3   | 8.0000E-02 : 6.5678E-02 : 7.1309E-02 : 6.5509E-02 : 7.1306E-02 : 6.3366E-02 :
4       | 4      | T4   | 8.0000E-02 : 6.5819E-02 : 5.2655E-02 : 4.2124E-02 : 3.7225E-02 : 3.4805E-02 :
5       | 5      | T5   | 8.0000E-02 : 7.2886E-02 : 6.0233E-02 : 4.8187E-02 : 5.0442E-02 : 5.4060E-02 :
6       | 6      | T6   | 8.0000E-02 : 7.8117E-02 : 7.2950E-02 : 6.9732E-02 : 7.4214E-02 : 7.5995E-02 :
7       | 7      | T7   | 8.0000E-02 : 8.3734E-02 : 8.75478E-02 : 9.5598E-02 : 9.6369E-02 : 9.7185E-02 :
8       | 8      | T8   | 8.0000E-02 : 9.4436E-02 : 1.1332E-01 : 1.3599E-01 : 1.3745E-01 : 1.3842E-01 :
9       | 9      | T9   | 8.0000E-02 : 9.6000E-02 : 1.1540E-01 : 1.3847E-01 : 1.4329E-01 : 1.4723E-01 :
10      | 10     | T10  | 8.0000E-02 : 9.6000E-02 : 1.1520E-01 : 1.3824E-01 : 1.4108E-01 : 1.4347E-01 :
*** USER INFORMATION MESSAGE 6464 (DOM12E)
RUN TERMINATED DUE TO HARD CONVERGENCE TO AN OPTIMUM AT CYCLE NUMBER = 5.

```



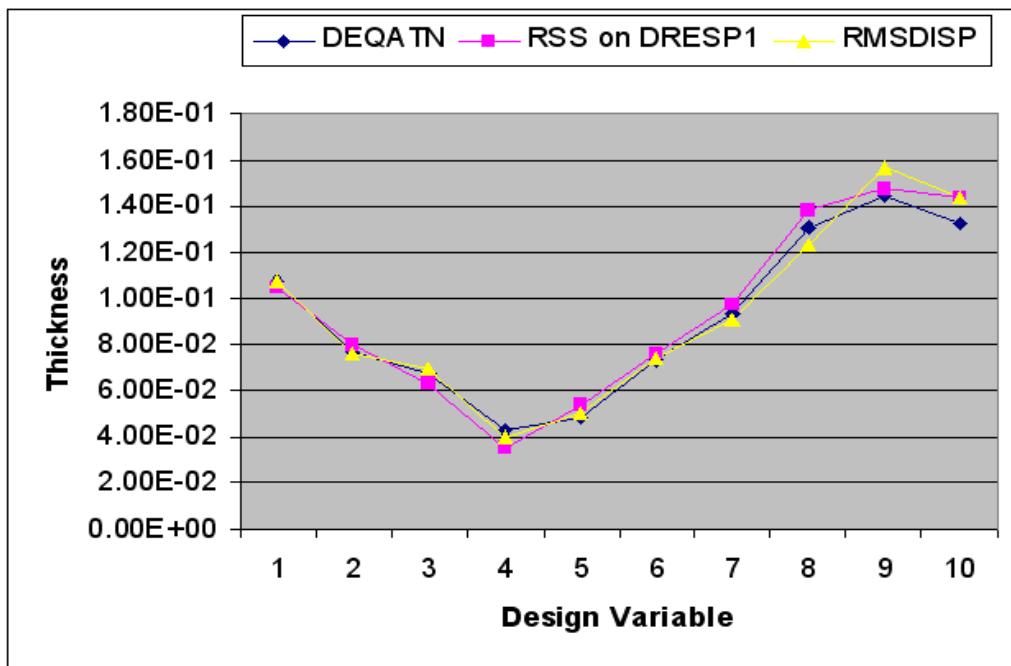


Figure 8-23 Final Thickness Distributions

Figure 8-24 shows the initial and final response data for the displacements used to formulate the objective for the first variation of this problem (the other variations have a similar form). 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 3.623 at 58 Hz to a value of 3.017 at 53 Hz, indicating a slight decrease in overall structural stiffness.



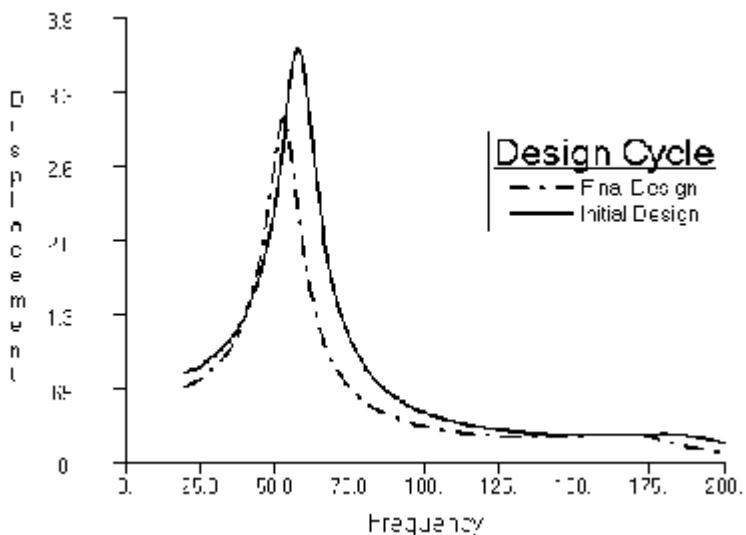


Figure 8-24 Frequency-Dependent Displacements



Twenty-Five Bar Truss, Superelement and Discrete Variable Optimization

This problem, often seen in the early 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 8-25](#). 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 and the final design will be selected from a user specified list of discrete variables.

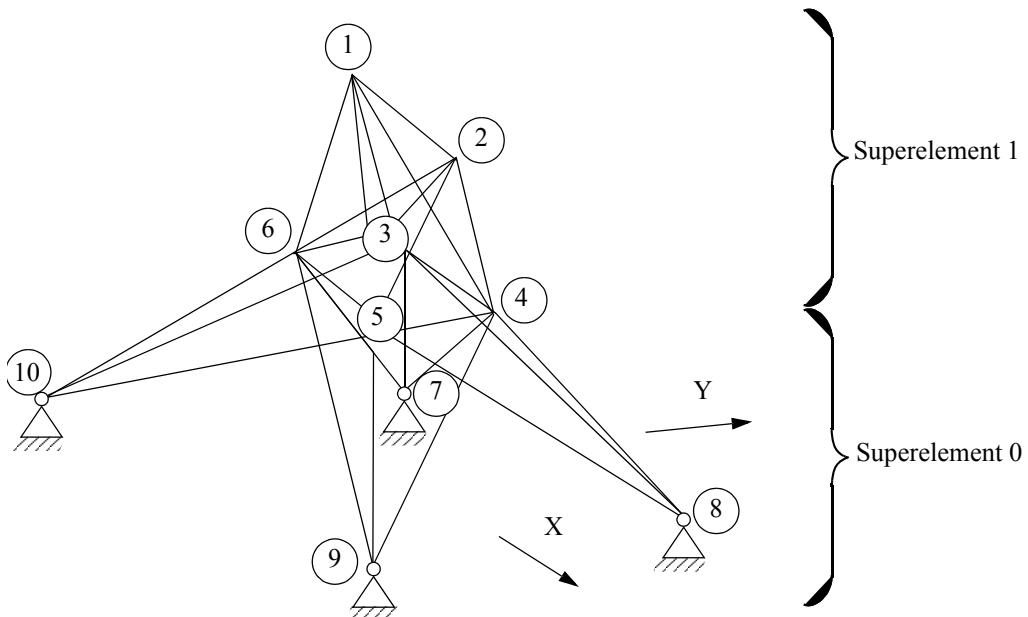


Figure 8-25 Twenty-Five Bar Truss

Analysis Model Description

Three-dimensional truss tower

Symmetric with respect to x-y plane and y-z plane

Weight density = 0.1/in³

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: ± 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 10

Euler buckling occurs when the magnitude of a member's compressive stress is greater than a critical stress which, for the first buckling mode of a pin-connected member, is

$$\sigma_b = \frac{P_b}{A} = \frac{1}{A} \left(-\frac{\pi^2 EI}{L^2} \right) \quad (8-8)$$

We will prescribe thin-walled tubular members with a diameter-to-thickness ratio of 10, as indicated in the sketch in [Figure 8-26](#).

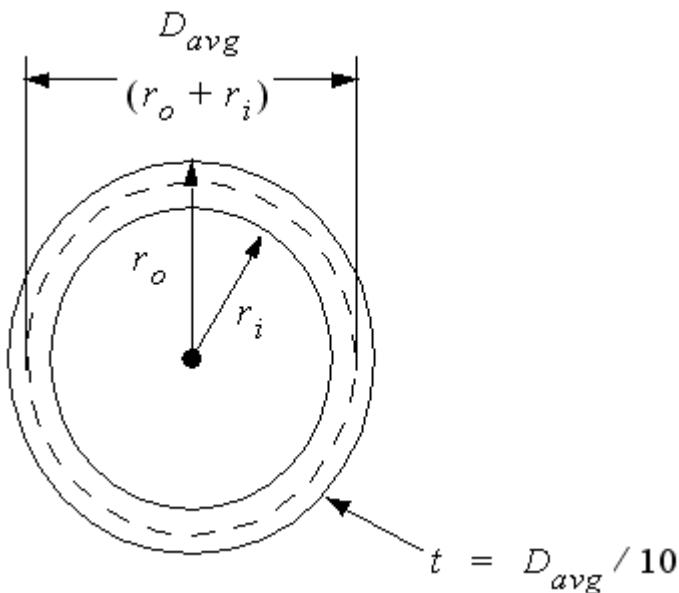


Figure 8-26 Thin-Walled Tube Cross-Section



With this formulation, the area and inertia terms of [Equation \(8-8\)](#) become

$$A = 2\pi(r_o^2 - r_i^2) = \pi D_{avg} t = \frac{\pi(D_{avg})^2}{10} \quad (8-9)$$

$$I = \frac{\pi}{4}(r_o^4 - r_i^4) = 0.013 \pi D_{avg}^4 \quad (8-10)$$

The critical buckling stress of [Equation \(8-8\)](#) can therefore be written as:

$$\sigma_b = -\frac{\pi^2 E}{L^2 A} I = -\frac{0.13 \pi^2 E D_{avg}^2}{L^2} \quad (8-11)$$

It is also desirable to impose a factor of safety on the design for buckling so that the buckling response constraint can be written as:

$$r = \frac{F_s \sigma}{\sigma_b} = \frac{-7.69 \sigma L^2 F_s}{\pi^2 E D_{avg}^2} \leq 1.0 \quad (8-12)$$

where F_s is the factor of safety.

The input data file for this example is shown in [Listing 8-17](#) and needs only limited description. Eight independent design variables (DESVAR 1 through 8) are used to represent eight ROD average diameters (the diameter of the midsurface of the thin-walled tube). A single list of discrete variables is invoked on each DESVAR entry to limit the allowable values of the design variables to 0.1, 0.5, or integer values between 1.0 and 100.0. DVPREL2 entries are used to relate the design variables to the rod areas using the relationship of [Equation \(8-9\)](#).

Listing 8-17 Input File for Example DSOUG8

```
ID MSC   DSOUG8 $
TIME 10
SOL 200      $  OPTIMIZATION
CEND
TITLE      = OPTIMAL SIZING OF A 25-BAR TRUSS  -
SUBTITLE   = EIGHT INDEPENDENT ROD DIAMETERS           DSOUG8
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 DEFINITION
SUBCASE 1
  LABEL = LOAD CONDITION 1
  LOAD = 300
SUBCASE 2
  LABEL = LOAD CONDITION 2
  LOAD = 310
BEGIN BULK
$-----
$----- ANALYSIS MODEL
$-----
```



```

$  

SESET 1 1 2  

GRDSET  

MAT1 1 10.0E6 0.1  

+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 DDVAL  

$  

DESVAR 1 D1 2.0 0.01 100.0 10  

DESVAR 2 D2 2.0 0.01 100.0 10  

DESVAR 3 D3 2.0 0.01 100.0 10

```



```

DESVAR 4      D4      2.0      0.01    100.0      10
DESVAR 5      D5      2.0      0.01    100.0      10
DESVAR 6      D6      2.0      0.01    100.0      10
DESVAR 7      D7      2.0      0.01    100.0      10
DESVAR 8      D8      2.0      0.01    100.0      10
DDVAL 10     0.1      0.5
              1.0      THRU     100.    BY      1.0
$...RELATE THE DESIGN VARIABLES TO THE ANALYSIS MODEL PROPERTIES
$DVPREL2 ID   TYPE   PID   FID   PMIN   PMAX   EQID
$      DESVAR DVID1
$DVPREL2 1    PROD    1     A      100
DESVAR 1
$DVPREL2 2    PROD    2     A      100
DESVAR 2
$DVPREL2 3    PROD    3     A      100
DESVAR 3
$DVPREL2 4    PROD    4     A      100
DESVAR 4
$DVPREL2 5    PROD    5     A      100
DESVAR 5
$DVPREL2 6    PROD    6     A      100
DESVAR 6
$DVPREL2 7    PROD    7     A      100
DESVAR 7
$DVPREL2 8    PROD    8     A      100
DESVAR 8
$...EQUATIONS USED TO DEFINE TYPE 2 PROPERTIES
$DEQATN EQUID  F() = ...
$DEQATN 100    AREA(DAVG) = PI(1) * DAVG**2 / 10.0;
$...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     123
DRESP1 12   D4      DISP     123
DRESP1 15   W       WEIGHT
$...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
DESVAR 1
DTABLE  FS     E      L1
DRESP1 1
$DRESP2 17   SC2    1
DESVAR 2
DTABLE  FS     E      L2
DRESP1 2
$DRESP2 18   SC3    1

```



```

DESVAR 3
DTABLE FS E L3
DRESP1 3
DRESP2 19 SC4 1
DESVAR 4
DTABLE FS E L4
DRESP1 4
DRESP2 20 SC5 1
DESVAR 5
DTABLE FS E L5
DRESP1 5
DRESP2 21 SC6 1
DESVAR 6
DTABLE FS E L6
DRESP1 6
DRESP2 22 SC7 1
DESVAR 7
DTABLE FS E L7
DRESP1 7
DRESP2 23 SC8 1
DESVAR 8
DTABLE FS E L8
DRESP1 8
$ . . . EQUATIONS USED TO DEFINE SECOND LEVEL RESPONSES
$ DEQATN EQUID F() = ...
$ . . .
DEQATN 1      NUM(DAVG,FS,E,L,SIGMA) = 7.69 * L**2 * SIGMA;
               DENOM           = (PI(1) * DAVG)**2 * E;
               BUCKLING         = -FS * NUM / DENOM
$ . . .
$ . . . 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
       E 1.0E7 FS 1.25
$ . . .
$ . . . 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 11 16 1.0
DCONSTR 11 17 1.0
DCONSTR 11 18 1.0
DCONSTR 11 19 1.0
DCONSTR 11 20 1.0
DCONSTR 11 21 1.0
DCONSTR 11 22 1.0
DCONSTR 11 23 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 1PRINT   3 DESMAX      15      DELP    0.5      P1      1
          P2      15
ENDDATA
$.....2.....3.....4.....5.....6.....7.....8.....9.....0

```

Axial stresses and grid displacements are identified on DRESP1 entries 1 through 12 and are constrained using DCONSTR entries with an ID 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. The DTABLE entry also defines the factor of safety at 125%. DCONSTR entries place bounds on the buckling response. Since the lower bound is not of interest (e.g., tensile stresses on the elements will not induce Euler buckling), we leave the lower bound blank on the DCONSTR entry.

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 WEIGHT response of DRESP1 ID 15 has an "ALL" to indicate the total structural weight across all superelements is to be computed and used as the objective Function. "ALL" is the default for this attribute so that even this adjustment in the design model to accommodate superelements was not strictly necessary.

Final results for the design task are shown in [Listing 8-18](#). Note that the design required five finite analyses: an initial design, three continuous redesigns, and a final discrete design. The initial design had a weight of 795.9 and was infeasible by a large amount (the maximum constraint value is 6.12). After eight redesigns, the weight increased to 814.1 and the design is feasible. A single discrete variable optimization is performed and the weight increases to 1007.3 while the maximum constraint decreases. The rather large increase in the objective of over 200. caused by the discrete design may motivate consideration of additional discrete sizes.

Listing 8-18 Design History Results for DSOUG8

```

*****
SUMMARY OF DESIGN CYCLE HISTORY
*****
(HARD CONVERGENCE ACHIEVED)
(SOFT FEASIBLE DISCRETE DESIGN OBTAINED)
(HARD FEASIBLE DISCRETE DESIGN OBTAINED)
NUMBER OF FINITE ELEMENT ANALYSES COMPLETED         9
NUMBER OF OPTIMIZATIONS W.R.T. APPROXIMATE MODELS    7
NUMBER OF DISCRETE PROCESSING ANALYSES COMPLETED     1
OBJECTIVE AND MAXIMUM CONSTRAINT HISTORY
-----
          OBJECTIVE FROM          OBJECTIVE FROM          FRACTIONAL ERROR          MAXIMUM VALUE
CYCLE    APPROXIMATE          EXACT              OF                      OF
NUMBER   OPTIMIZATION        ANALYSIS           APPROXIMATION        CONSTRAINT
-----
```

CYCLE	APPROXIMATE	EXACT	OF	OF
NUMBER	OPTIMIZATION	ANALYSIS	APPROXIMATION	CONSTRAINT
INITIAL		4.155959E+02		6.115134E+00
1	8.312938E+02	8.313505E+02	-6.820427E-05	4.031373E-01
2	8.072191E+02	8.071776E+02	5.134294E-05	1.130664E-02
3	8.008032E+02	8.008042E+02	-1.295695E-06	7.897258E-03
4	7.992347E+02	7.992356E+02	-1.069137E-06	7.599592E-04



```

5      7.980751E+02      7.980754E+02      -3.823897E-07      6.883144E-04
6      7.972520E+02      7.972520E+02      0.000000E+00      6.126165E-04
7      7.972520E+02      7.972520E+02      0.000000E+00      6.126165E-04
7D     1.007326E+03      1.007323E+03      2.484249E-06      -1.106629E-01

-----
1  OPTIMAL SIZING OF A 25-BAR TRUSS  -          DSOUG8        SEPTEMBER 7, 2009 MSC NASTRAN 9/ 4/09 PAGE 228
EIGHT INDEPENDENT ROD DIAMETERS
0
1  OPTIMAL SIZING OF A 25-BAR TRUSS  -          DSOUG8        SEPTEMBER 7, 2009 MSC NASTRAN 9/ 4/09 PAGE 229
EIGHT INDEPENDENT ROD DIAMETERS
0

DESIGN VARIABLE HISTORY
-----
INTERNAL | EXTERNAL |           |
DV. ID. | DV. ID. | LABEL | INITIAL : 1 : 2 : 3 : 4 : 5 :
-----
1 | 1 | D1 | 2.0000E+00 : 1.4208E+00 : 1.0047E+00 : 7.1041E-01 : 5.0231E-01 : 4.7024E-01 :
2 | 2 | D2 | 2.0000E+00 : 2.9919E+00 : 2.9663E+00 : 2.9540E+00 : 2.9488E+00 : 2.9474E+00 :
3 | 3 | D3 | 2.0000E+00 : 2.7639E+00 : 2.8707E+00 : 2.8763E+00 : 2.8829E+00 : 2.8837E+00 :
4 | 4 | D4 | 2.0000E+00 : 1.6213E+00 : 1.1465E+00 : 8.1066E-01 : 5.7323E-01 : 4.0534E-01 :
5 | 5 | D5 | 2.0000E+00 : 1.2827E+00 : 1.1656E+00 : 1.1853E+00 : 1.1926E+00 : 1.1925E+00 :
6 | 6 | D6 | 2.0000E+00 : 2.9880E+00 : 2.5028E+00 : 2.4939E+00 : 2.4925E+00 : 2.4880E+00 :
7 | 7 | D7 | 2.0000E+00 : 2.9919E+00 : 3.2856E+00 : 3.2867E+00 : 3.2912E+00 : 3.2922E+00 :
8 | 8 | D8 | 2.0000E+00 : 2.9919E+00 : 2.9805E+00 : 2.9735E+00 : 2.9733E+00 : 2.9744E+00 :

INTERNAL | EXTERNAL |           |
DV. ID. | DV. ID. | LABEL | 6 : 7 : 7D : 8 : 8D : 9 :
-----
1 | 1 | D1 | 4.0382E-01 : 4.0382E-01 : 1.0000E-01 :
2 | 2 | D2 | 2.9446E+00 : 2.9446E+00 : 3.0000E+00 :
3 | 3 | D3 | 2.8854E+00 : 2.8854E+00 : 3.0000E+00 :
4 | 4 | D4 | 2.8662E-01 : 2.8662E-01 : 1.0000E-01 :
5 | 5 | D5 | 1.1943E+00 : 1.1943E+00 : 2.0000E+00 :
6 | 6 | D6 | 2.4853E+00 : 2.4853E+00 : 3.0000E+00 :
7 | 7 | D7 | 3.2933E+00 : 3.2933E+00 : 4.0000E+00 :
8 | 8 | D8 | 2.9742E+00 : 2.9742E+00 : 3.0000E+00 :

*** USER INFORMATION MESSAGE 6464 (DOM12E)
RUN TERMINATED DUE TO HARD CONVERGENCE TO AN OPTIMUM AT CYCLE NUMBER = 7.
AND HARD FEASIBLE DISCRETE DESIGN OBTAINED

```



Design Optimization with Composite Materials with Fully Stressed Design

In the optimal design of laminated composite structures, individual ply thicknesses and orientations are often selected as design quantities. [Figure 8-27](#) shows a simple wing model that has an aluminum substructure and wing cover skins that are made from composite materials. These cover skins have layers in the 0,90,+45 and -45 degree directions relative to material coordinate system that is aligned with the 50% chord of the wing structure. Variations of this model have appeared numerous times in the design optimization literature. Johnson and Neill provide a representative description (see Reference 14.).

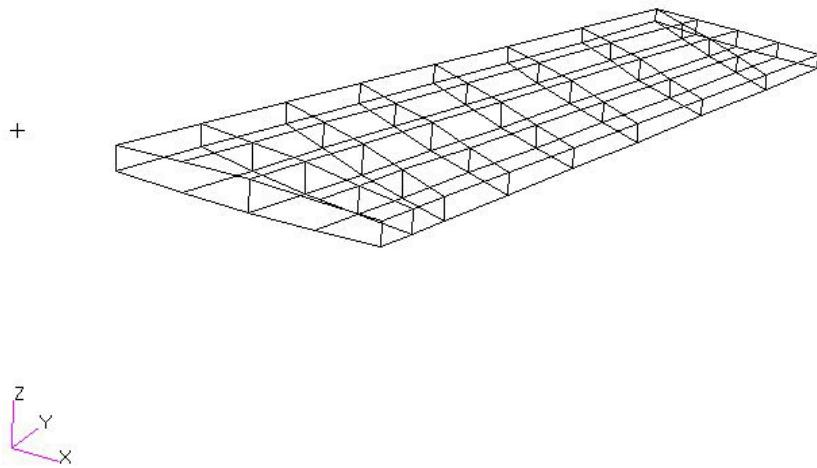


Figure 8-27 Intermediate Complexity Wing Model

In the variation of the design task given here, the wing structure is subjected to two static load cases and a normal modes analysis is also performed. The Fully Stressed Design (FSD) technique is employed to find a good starting design for the complete design task, which requires a mathematical programming approach. As discussed in [Fully Stressed Design](#), the FSD technique can only deal with static loads and a limited subset of design responses and designed properties. The design task is to minimize the wing's structural weight while satisfying the following strength and stiffness considerations:



Composite Plies	Strain is below failure point (see the CFAILURE discussion on page 556).
Posts in Substructure	$-5.7 \cdot 10^4 \leq \sigma \leq 6.7 \cdot 10^4$
Webs in Substructures	$-3.9 \cdot 10^4 \leq \sigma \leq 3.9 \cdot 10^4$
Second Natural Frequency	$f_2 \geq 40. \text{ Hz}$

The input data file is much too large to be listed here in its entirety so that only fragments are shown here in [Listing 8-19](#). The complete input file can be found in the MSC Nastran TPL with the name dsoug9.dat. The finite element model consists of 109 GRIDSS, 39 CRODs that link top and bottom structural grids, 55 CSHEARs that model the wing ribs and spars and 2 CTRIA3s and 62 CQUAD4s that model the upper and lower wing skins. 39 CONM2s model the nonstructural mass of the wing and 102 MPCs are used to connect the grids associated with the mass model structural model.

The design model contains 153 DESVARs. A single DESVAR controls the areas of all the rods in the model while a second DESVAR controls the thickness of all the ribs. The thicknesses of each of the 23 CSHEARs that model the spars are independently designed. The remaining 128 design variables control the thicknesses of the four layers of the composite cover skins. For the design concept employed in this example, unequal number of plus and minus 45 degree plies are allowed but the top and bottom surfaces are constrained to have the same thickness values. Each of the 32 finite elements that model a surface is independently designed. Clearly, there are many ways to specify the design model for this structure and the one shown may not be the most practical or desirable. You may wish to try decoupling the top and bottom surfaces for example or to see what the weight penalty is for imposing a restriction that the number of +45 plies must equal the number of -45 plies.

Analysis and design with composite materials is a advanced feature in MSC Nastran and it is felt worthwhile to point out some of the special features that were used in this example. The first is that the SYM option has been used for the LAM attribute on the PCOMP Bulk Data entry. With this option, it is only necessary to model half the layup with the other half added in such a way that there is a equal amount of material above and below the midplane of the laminate. In an actual layup, it would be necessary to specify the order of the plies in terms of individual laminate. This ordering is not possible here and it is seen that the zero plies are all lumped at the extreme locations in the laminate with the 90, +45 and -45 directions following so that the -45 plies are at the midplane.

The objective is to minimize the weight and a DRESP2 is used to subtract 115. pounds from the weight computed using the DRESP1 with RTYPE=WEIGHT. The 115. value is a an estimate of the weight of the concentrated masses that is not affected by changes in design. Removing this fixed weight from the optimizer allows the algorithm to focus on the weight it can affect.

The DVPREL1's that specify the ply thicknesses multiply the DESVAR value by .00125. The intent here is to make the DESVAR value a count of the required number of plies under the assumption that each laminate has a thickness of .00250. Discrete variables could have been used in this example to make the design variables an integer ply count, but this was not done.

It is also necessary to take care in specifying the location of the bottom surface of the laminate (Z0 on the PCOMP entry). The grid locations are taken to specify the OML (outer mold line) of the wing that



has been specified to achieve the desired aerodynamic shape and performance. The structural material must therefore lie within the OML so that Z0 must be designed to be equal to the total thickness of the laminate. Also note that the direction of the bottom surface is governed by the order in which the element is numbered so that care must be taken to make this order consistent with the design intent. The Z0 quantity must be designed using a DVPREL1 entry so that the bottom surface changes as the individual plies change. Note that the -.0025 factor accounts for the fact that the total thickness is twice the thickness given on the PCOMP entry.

Listing 8-19 Input File Fragments for DSOUG9

```
ID MSC, DSOUG9 $ v2004 ehj 25-Jun-2003
TIME 200 $
SOL 200 $
CEND
TITLE =INTERMEDIATE COMPLEXITY WING **STAT & EIGN CONSTRAINTS* DSOUG9
SUBIT = QUAD4 ELEMENTS WITH 153 DESIGN VARIABLES ** CMS ***
LABEL = COMPOSITE STRUCTURE WITH FIBER ORIENTATIONS (0,90,+45,-45)
ECHO = SORT
SPC = 1
MPC = 200
DESOBJ= 1001
DESSUB= 1
SUBCASE 1
    ANALYSIS=STATIC
    METHOD=10
    LOAD = 1
SUBCASE 2
    ANALYSIS=STATIC
    METHOD=10
    LOAD = 2
SUBCASE 3
    DESSUB = 3
    ANALYSIS=MODES
    METHOD=10
BEGIN BULK
$
DCONSTR 1      1      1.0
DCONSTR 1      2      1.0
DCONSTR 1      3      1.0
DCONSTR 1      4      1.0
DCONSTR 1     11      1.0
DCONSTR 1     12      1.0
DCONSTR 1     13      1.0
DCONSTR 1     14      1.0
DCONSTR 1     21   -5.7+4  6.7+4
DCONSTR 1     31   -3.9+4  3.9+4
DCONSTR 3     200     40.0
DESVAR 33      RIBS  0.21  0.20
DESVAR 34      SHEAR1 0.21  0.20
DESVAR 57      POSTS 0.21  0.20
DESVAR 1101    CTRIA31 1.000 1.000000
DESVAR 1102    CQUAD1  1.001 1.000000
DESVAR 1201    TRIA31  1.001 1.000000
DESVAR 1432    CQUAD32 1.001 1.000000
DOPTPRM FSDMAX 10      DESMAX 20      DELP   0.50      DPMIN  0.001
              delx  0.49    p1     1      p2     15
DRESP1 1       STRAIN1 CFFAILUREPCOMP 5      1      10001
DRESP1 2       STRAIN1 CFFAILUREPCOMP 5      2      10001
DRESP1 3       STRAIN1 CFFAILUREPCOMP 5      3      10001
```



```

DRESP1 4      STRAIN1 CFAILUREPCOMP          5      4      10001
DRESP1 11     STRAIN1 CFAILUREPCOMP          5      1      30031
            30001 30002 30003 30004 30005 30006 30007 30008
            30009 30010 30011 30012 30013 30014 30015 30016
            30017 30018 30019 30020 30021 30022 30023 30024
            30025 30026 30027 30028 30029 30030
DRESP1 12      STRAIN1 CFAILUREPCOMP          5      2      30031
            30001 30002 30003 30004 30005 30006 30007 30008
DRESP1 21      STRESS  STRESS  PROD           2      10001
DRESP1 31      STRESS  STRESS  PSHEAR          3      40001
            40002 40003 40004 40005 40006 40007 40008 40009
            40010 40011 40012 40013 40014 40015 40016 40017
            40018 40019 40020 40021 40022 40023 40024
DRESP1 101     W      WEIGHT                ALL
DRESP2 1001    STRWGT 1001
DRESP1 101
DEQATN 1001   STRWGT(TOTAL) = TOTAL - 115.
DRESP1 200     FREQ   FREQ               2      1
DVPREL1 33     PSHEAR 40001  T       .02
            33     0.1
DVPREL1 57     PROD   10001  A       .0001
            57     0.1
DVPREL1 1101   PCOMP  10001  T1
            1101   0.00125
DVPREL1 2432   PCOMP  32031  T4
            1432   0.00125
DVPREL1 4101   PCOMP  10001  Z0
            1101   -.00250 1201  -.00250 1301  -.00250 1401  -.00250
DVPREL1 5132   PCOMP  32031  Z0
            1132   -.00250 1232  -.00250 1332  -.00250 1432  -.00250
DYNRED 100    300.0
EIGRL 10      4      0
FORCE 1       1      0      1.0    205.0   -7380.0 926.0
FORCE 1       2      0      1.0    -205.0   7380.0 926.0
FORCE 2       78     0      1.0    451.0   -86.0   175.0
GRID 1        63.5000 90.00000 1.1250
GRID 2        63.5000 90.00000 -1.1250
GRID 217     85.5    90.000  0.0
MAT1 10       1.05E+7 4.04E+6 0.30000 0.10000 0.00000 0.00000 0.00000
            6.70E+4 5.70E+4 3.90E+4
MAT8 70       1.85E+7 1.60E+6 0.25000 0.65E6          0.05500
            0.0     0.0    100.  1.15E+5 1.15E+5 1.15E+5 1.15E+5 1.0E+15
MAT8 72       1.85E+7 1.60E+6 0.25000 0.65E6          0.05500
            0.0     0.0    100.
MPC 200      1       1      1.0    201     1      -.8924
            201     5     -1.0040 201     2      -.4512
            201     4     .5076
MPC 200      1       2      1.0    201     2      -.8924
            201     1     .4512 201     5     .5076
            201     4     1.0040
PARAM WTMASS 0.00259
PARAM POST   -1
PCOMP 10001   -.0105 0.0    0.65E6  TSAI          SYM
            70     0.500 0.0    YES    70     0.500 90.  YES
            70     0.500 45.   YES    70     0.500 -45.  YES
PCOMP 32031   -.0105 0.0    YES    72     0.500 90.  YES
            72     0.500 0.0    YES    72     0.500 -45.  YES
PROD 10001   10     1.00
PSHEAR 40001  10     1.0
PSHEAR 40024  10     1.0
SPC1 1       6     201    THRU   217

```



```
SPC1      1      123456   79      THRU     88
ENDDATA
```

Finally, the design task specifies a constraint on a CFAILURE response. The correct specification of this response requires a combination of inputs that are described here. On the DRESP1, the CFAILURE response type is invoked for each of the designed properties for each of the PCOMPS on the upper surface. Item code 5 is specified, which from [Table 6-2](#) in the *MSC Nastran Quick Reference Guide* is seen to be a failure mode for direct stresses. The type of the failure mode is specified as TSAI (for Tsai-Wu theory) on the PCOMP entry. [Table 13-1](#) in [Composites](#) in Chapter 13 of the *MSC Nastran Reference Manual* gives the formula for the Tsai-Wu criteria as:

$$\left(\frac{1}{X_t} - \frac{1}{X_c}\right)\sigma_1 + \left(\frac{1}{Y_t} - \frac{1}{Y_c}\right)\sigma_2 + \frac{\sigma_1^2}{X_t X_c} + \frac{\sigma_2^2}{Y_t Y_c} + \frac{\sigma_{12}^2}{S^2} + 2F_{12}\sigma_1\sigma_2 = FI$$

The X_t , X_c , Y_t , Y_c , S and F_{12} values are specified on the material entry, MAT8 in this example.

The DOPTPRM Bulk Data entry is used to specify that up to 10 FSD cycles are to be performed followed by up to 20 mathematical programming cycles, where “up to” refers to the fact that both the FSD and MP algorithms will terminate before the specified number of cycles if the design is judged to have converged.

[Listing 8-20](#) shows design history data for the example while [Figure 8-28](#) shows some of the same data in xy plot form. It is seen from [Listing 8-20](#) that ten cycles of FSD were performed followed by six MP cycles. The FSD algorithm did an effective job of coming up with a good starting design for the MP algorithm in that the maximum constraint value is reduced from 570 to near 1 (note the log scale in [Listing 8-20](#)). The final FSD design has a weight of 75.16 lbs but is infeasible. The final design is feasible and has a weight of 78.1 lbs. For this simple example, the CPU time for each design cycle was approximately 2 secs. for each FSD cycles and 8 secs. for each MP cycle, illustrating the performance advantage of the FSD method.

Listing 8-20 Design Cycle History for DSoug9

```
*****
* SUMMARY OF DESIGN CYCLE HISTORY
*****
(HARD CONVERGENCE ACHIEVED)
NUMBER OF FINITE ELEMENT ANALYSES COMPLETED          21
NUMBER OF FULLY STRESSED DESIGN CYCLES COMPLETED      10
NUMBER OF OPTIMIZATIONS W.R.T. APPROXIMATE MODELS    10
OBJECTIVE AND MAXIMUM CONSTRAINT HISTORY
-----
CYCLE      OBJECTIVE FROM APPROXIMATE           OBJECTIVE FROM EXACT           FRACTIONAL ERROR OF APPROXIMATION           MAXIMUM VALUE OF CONSTRAINT
NUMBER     OPTIMIZATION                   ANALYSIS
-----
INITIAL
1          FSD                         5.392062E+01
2          FSD                         5.785173E+01
3          FSD                         6.312259E+01
4          FSD                         6.776628E+01
5          FSD                         7.179587E+01
6          FSD                         7.476866E+01
7          FSD                         7.571889E+01
8          FSD                         7.566803E+01
9          FSD                         7.587285E+01
                                7.525336E+01
N/A
```



10	FSD	7.413710E+01	N/A	2.869036E+00
11	7.983705E+01	7.983514E+01	2.389109E-05	4.608449E-01
12	7.680142E+01	7.680116E+01	3.377545E-06	1.316648E+00
13	7.853728E+01	7.853757E+01	-3.691443E-06	1.649760E-01
14	7.819966E+01	7.819945E+01	2.731772E-06	7.861853E-04
15	7.754909E+01	7.754898E+01	1.377343E-06	7.438540E-03



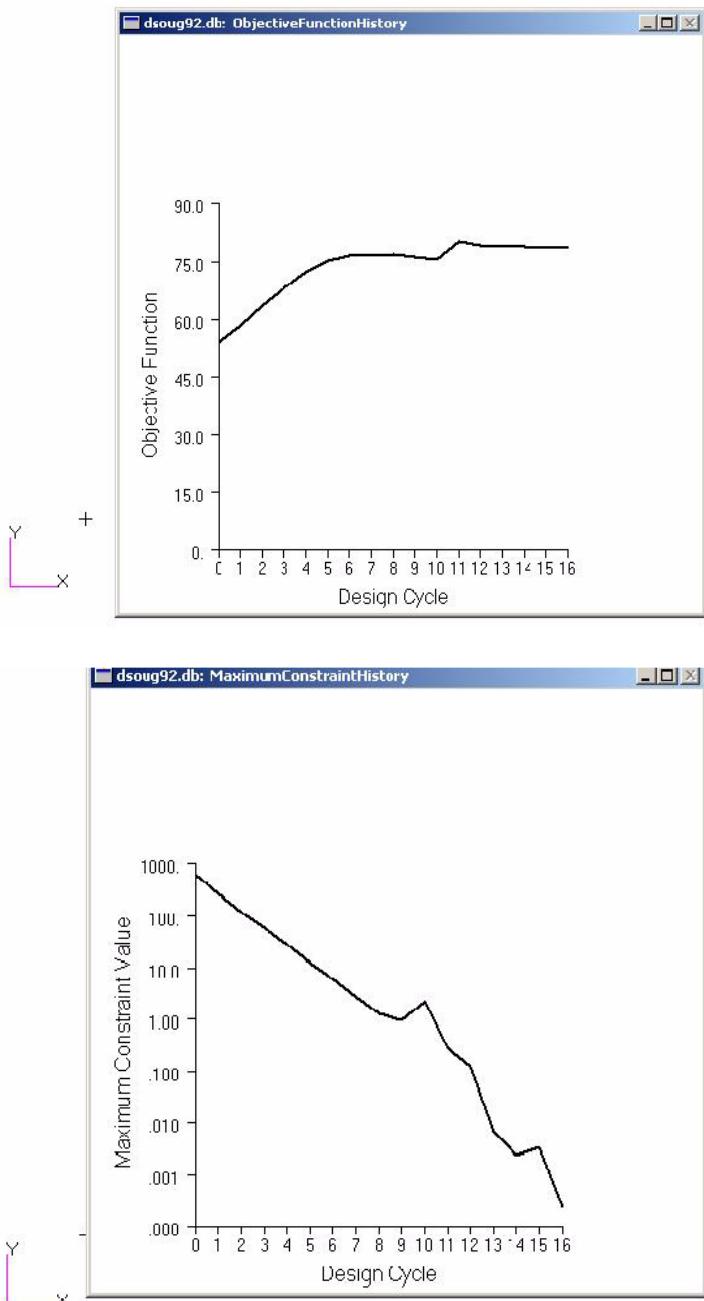


Figure 8-28 Selected Design Cycle Results for DSOUG9

Acoustic Optimization

Acoustic Optimization uses 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 without increasing the weight of the box.

The box geometry and property groups of thicknesses to be modified are shown in [Figure 8-29](#). Six design variables are to be related to six of these property groups (the third property group in [Figure 8-29](#) remains fixed.) The model consists of 1000 structural elements and 2000 fluid elements.

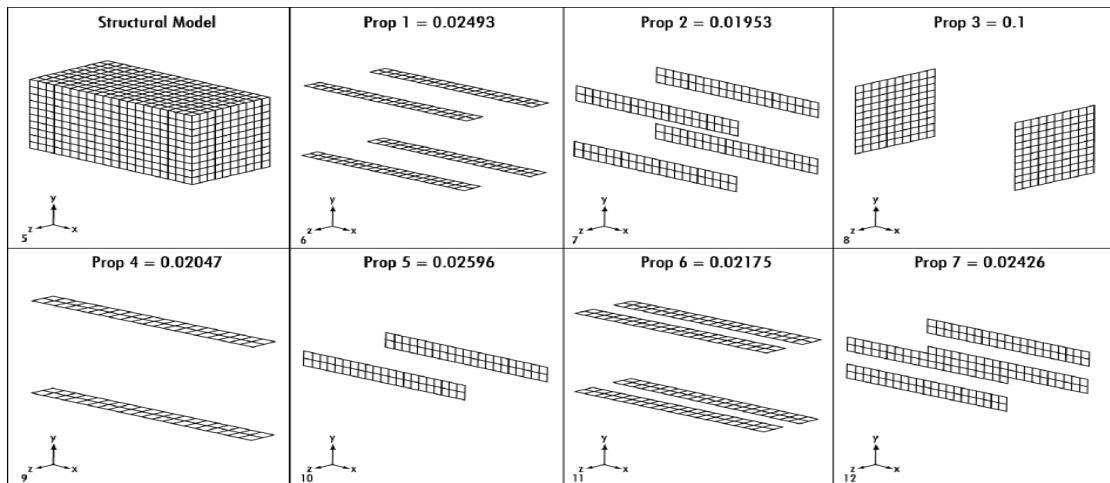


Figure 8-29 Acoustic Box Showing Portions Designed by Each Design Variable
(Prop 3 is Fixed)

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 8-30](#) 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.

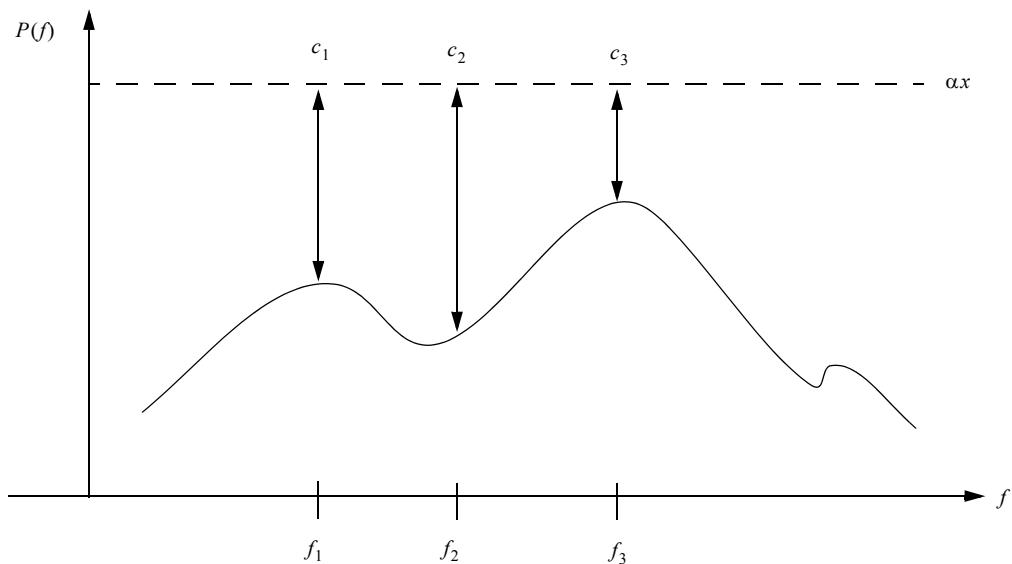


Figure 8-30 Minimization of Response Peaks

We can choose a design variable to represent a peak threshold, here shown in the figure as βx (is just a constant of proportionality to facilitate scaling the threshold). The difference between βx and the pressure distribution must be a positive quantity at all frequencies of interest. Thus, constraints can be written which require c_1 , c_2 , and c_3 to be positive distances. The design objective can then be to minimize βx , or

minimize

$$\beta x \quad (8-13)$$

subject to:

$$\begin{aligned} P(f_1) - \beta x &\leq 0 \\ P(f_2) - \beta x &\leq 0 \\ P(f_3) - \beta x &\leq 0 \end{aligned} \quad (8-14)$$

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 8-21](#) 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 [Additional Topics](#) of the *MSC Nastran Reference Guide*.



Listing 8-21 Fragments from Input Data File DSOUG10

```

ID MSC, DSOUG10 $ v2004 ehj 25-Jun-2003
$ Modified 26-Jul-2005 v2005 ehj
TIME 9999
SOL 200 $ MODAL FREQUENCY RESPONSE
CEND
TITLE = DESIGN OPTIMIZATION WITH ACOUSTICS
SUBTITLE = ACOUSTIC AND STRUCTURAL ELEMENTS
LABEL = BOXAE1.DAT
SET 20 = 11280
$-----2-----3-----4-----5-----6-----7-----8-----9-----10-----
ECHO = SORT(PARAM,EIGC,EIGRL,FREQ,DESVAR,DCONSTR,DRESP1,DRESP2,DEQATN,
             DVPREL1)
SPC      = 1
DESGLB   = 5
DISP(PHASE) = 20
METHOD(STRUCTURE) = 20
METHOD(FLUID)    = 30
$
FREQUENCY = 200
DLOAD     = 100
DESSUB    = 10
DESOBJ    = 100
ANALYSIS  = MFREQ
$
BEGIN BULK
$
$-----2-----3-----4-----5-----6-----7-----8-----9-----10-----
EIGRL 20          300.
EIGRL 30          15.       200.           0            105.
$
$ 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
$
PARAM POST      -1
$-----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
$-----2-----3-----4-----5-----6-----7-----8-----9-----10-----
FREQ1 200    40.    4.0    41
FREQ   200    97.5   102.5
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$ SENSITIVITY
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$
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
$
DVPREL1,1,PSHELL,1,T,0.0001
,1,1.
DVPREL1,2,PSHELL,2,T,0.0001
,2,1.
DVPREL1,4,PSHELL,4,T,0.0001
,4,1.
DVPREL1,5,PSHELL,5,T,0.0001
,5,1.
DVPREL1,6,PSHELL,6,T,0.0001
,6,1.
DVPREL1,7,PSHELL,7,T,0.0001
,7,1.
$
DRESP2 100    BETA    100
DESVAR 8
DEQATN 100    OBJ(BETA) = 10000.0 * BETA
DRESP1 1      PRESS   FRDISP           1          11280
DRESP1 2      WEIGHT  WEIGHT
DCONSTR 5     2      2890.  2910.
DRESP2 11     PRESBET 10
DESVAR 8
DRESP1 1
DEQATN 10    F(BETA,PRES) = 100.0 * BETA - PRES + 100.
DCONSTR 10   11    100.
$
$-----2-----3-----4-----5-----6-----7-----8-----9-----10-----
$
$
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$
STRUCTURAL MODEL
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$
PSHELL 1      1      .02493  1
PSHELL 2      1      .01953  1
PSHELL 3      1      .100    1
PSHELL 4      1      .02047  1
PSHELL 5      1      .02596  1
PSHELL 6      1      .02175  1
PSHELL 7      1      .02426  1
$
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 998    3      1138   1139   296   317
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-----
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$      ACOUSTIC MODEL
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$-----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    11279      1.       .5       .1       -1
GRID    11280      1.       .5       0.0      -1
GRID    11281      1.       .6       1.       -1
GRID    12539      2.       1.       .3       -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.
DOPTPRM DESMAX 30      P1      1      P2      15      APRCOD 1
      GMAX 0.015      DELX 0.5      CONV1 .005      adscod 1
ENDDATA

```

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 using the DRESP2 number 100 entry. This refers to DEQATN 100 which defines

$$F = 10000 \cdot x_8 \quad (8-15)$$

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 response is 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:

$$k_1 \cdot x_8 - P(f) + k_2 \geq k_2 \quad (8-16)$$

Note that this is similar to [Equation \(8-14\)](#), 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. In this example, k_1 and k_2 are set at 100. This is in line with a nominal peak pressure.

[Listing 8-22](#) and [Figure 8-31](#), and [Figure 8-32](#) and [Figure 8-33](#) present optimization results. The design cycle history of [Listing 8-22](#) shows the objective has decreased from 10000.0 to 282.8 in thirteen iterations. A comparison of the plots of [Figure 8-31](#) and [Figure 8-32](#) indicates that although the objective appears to have reached its minimum after eleven design cycles, the design variables continue to change. This indicates that the design space is quite flat for this case, giving the designer some freedom in the selection of the design variable values. The benefit of the procedure is dramatically demonstrated in [Figure 8-33](#), which depicts a reduction in the peak pressure from 205 N/mm^2 to 3 N/mm^2 . The technique of minimizing the maximum pressure has resulted in frequency response function that has three peaks near the maximum value whereas the initial function has a single sharp peak.

As a final comment on this test case, it has been found the results achieved can be affected by the platform and version of MSC Nastran that is used to run the problem. It is conjectured that this test is particularly sensitive due to the fact that there are a number of repeated eigenvalues in the fluid eigenanalysis including 3 at an eigenfrequency of 100.412. If you exercise this test, the same qualitative behavior of a significant reduction in the peak pressure should be observed, but it will likely not give the same final objective or number of design cycles.

Listing 8-22 Design Cycle History for DSOUG10

***** 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	11	NUMBER OF OPTIMIZATIONS W.R.T. APPROXIMATE MODELS	10	
OBJECTIVE AND MAXIMUM CONSTRAINT HISTORY				
CYCLE NUMBER	OBJECTIVE FROM APPROXIMATE OPTIMIZATION	OBJECTIVE FROM EXACT ANALYSIS	FRACTIONAL ERROR OF APPROXIMATION	MAXIMUM VALUE OF CONSTRAINT
INITIAL	5.000000E+03	5.000000E+03	0.000000E+00	1.047196E+00
1	2.500000E+03	2.500000E+03	0.000000E+00	5.302348E-01
2	1.875000E+03	1.875000E+03	0.000000E+00	1.581139E+00
3	1.406250E+03	1.406250E+03	0.000000E+00	6.556612E-01
4	1.054688E+03	1.054688E+03	0.000000E+00	-2.664237E-03
5	7.910156E+02	7.910156E+02	0.000000E+00	-2.664237E-03
6				



```

7      5.410156E+02      5.410156E+02      0.000000E+00      -2.664069E-03
8      2.910156E+02      2.910156E+02      0.000000E+00      1.563026E-02
9      2.465884E+02      2.465884E+02      0.000000E+00      3.949394E-02
10     2.465884E+02      2.465884E+02      0.000000E+00      3.949394E-02
-----
1 DESIGN OPTIMIZATION WITH ACOUSTICS          SEPTEMBER 7, 2009 MSC NASTRAN 9/ 4/09
PAGE 285
ACOUSTIC AND STRUCTURAL ELEMENTS
0 BOXAE1.DAT
DESIGN VARIABLE HISTORY
-----
INTERNAL | EXTERNAL |           |           |           |           |           |           |           |           |           |           |
DV. ID. | DV. ID. | LABEL    | INITIAL   : 1       : 2       : 3       : 4       : 5       :
-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|
1 | 1 | P1 | 2.4930E-02 : 2.4055E-02 : 1.2332E-02 : 1.2820E-02 : 1.3768E-02 : 1.3768E-02 :
2 | 2 | P2 | 1.9530E-02 : 1.8461E-02 : 3.0069E-02 : 2.9661E-02 : 3.1808E-02 : 3.1808E-02 :
3 | 4 | P4 | 2.0470E-02 : 2.3174E-02 : 3.8763E-02 : 3.8351E-02 : 4.1256E-02 : 4.1256E-02 :
4 | 5 | P5 | 2.5960E-02 : 2.3612E-02 : 3.9067E-02 : 3.8490E-02 : 3.1648E-02 : 3.1648E-02 :
5 | 6 | P6 | 2.1750E-02 : 2.5321E-02 : 2.2363E-02 : 2.2720E-02 : 2.7157E-02 : 2.7157E-02 :
6 | 7 | P7 | 2.4260E-02 : 2.2454E-02 : 9.9959E-03 : 1.0053E-02 : 4.4859E-03 : 4.4859E-03 :
7 | 8 | BETA | 1.0000E+00 : 5.0000E-01 : 2.5000E-01 : 1.8750E-01 : 1.4062E-01 : 1.0547E-01 :
-----
INTERNAL | EXTERNAL |           |           |           |           |           |           |           |           |           |           |
DV. ID. | DV. ID. | LABEL    | 6       : 7       : 8       : 9       : 10      : 11      :
-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|
1 | 1 | P1 | 1.3768E-02 : 1.3750E-02 : 1.4111E-02 : 1.5029E-02 : 1.5029E-02 :
2 | 2 | P2 | 3.1808E-02 : 3.1829E-02 : 2.6829E-02 : 2.1829E-02 : 2.1829E-02 :
3 | 4 | P4 | 4.1256E-02 : 4.1429E-02 : 3.7737E-02 : 3.9672E-02 : 3.9672E-02 :
4 | 5 | P5 | 3.1648E-02 : 3.1414E-02 : 3.4455E-02 : 3.3552E-02 : 3.3552E-02 :
5 | 6 | P6 | 2.7157E-02 : 2.7312E-02 : 3.2309E-02 : 3.4802E-02 : 3.4802E-02 :
6 | 7 | P7 | 4.4859E-03 : 4.3595E-03 : 4.2995E-03 : 4.3957E-03 : 4.3957E-03 :
7 | 8 | BETA | 7.9102E-02 : 5.4102E-02 : 2.9102E-02 : 2.4659E-02 : 2.4659E-02 :

```

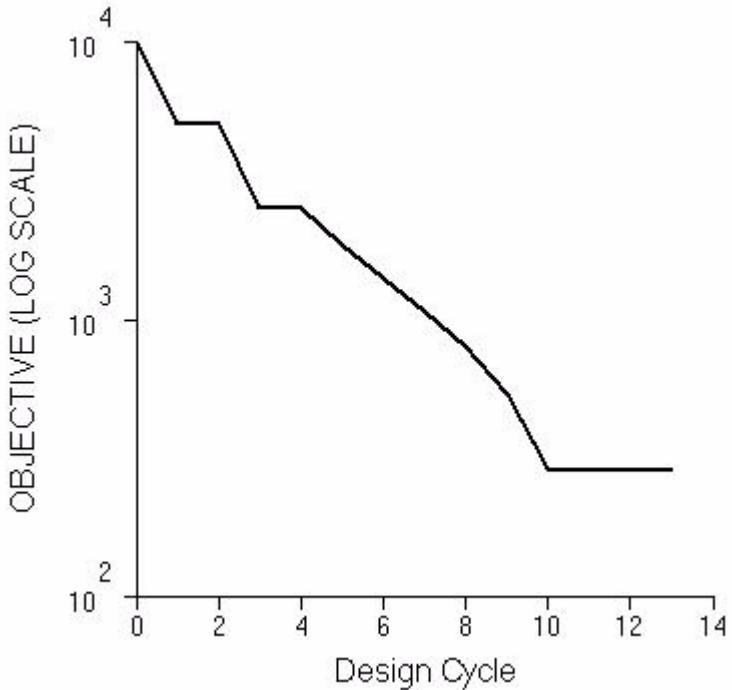


Figure 8-31 Objective Function versus Design Cycle



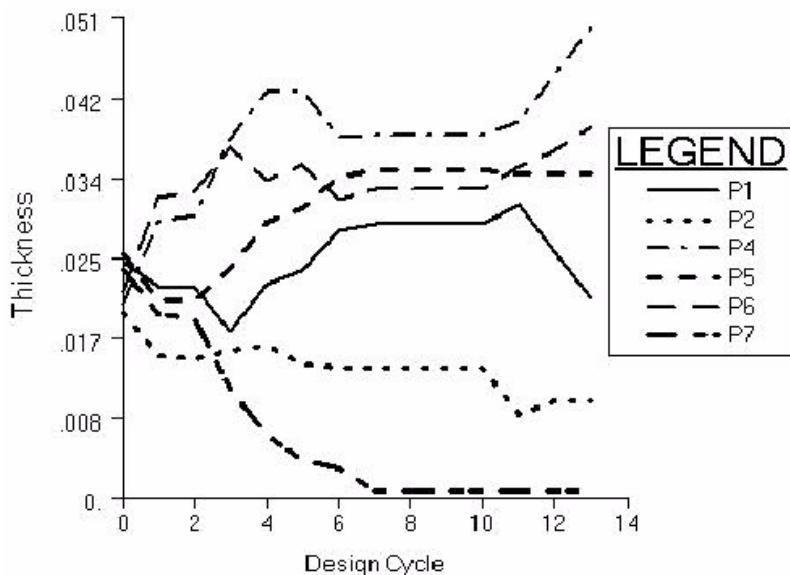


Figure 8-32 Design Variables versus Design Cycle

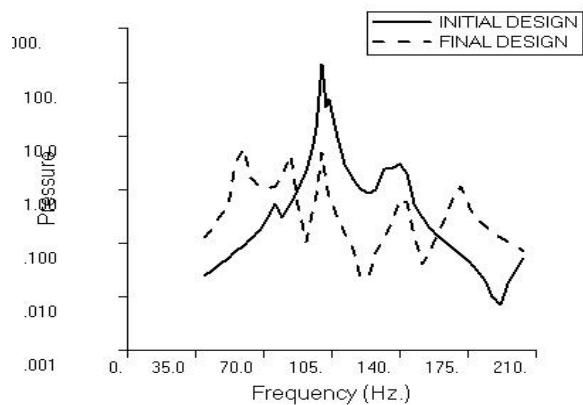
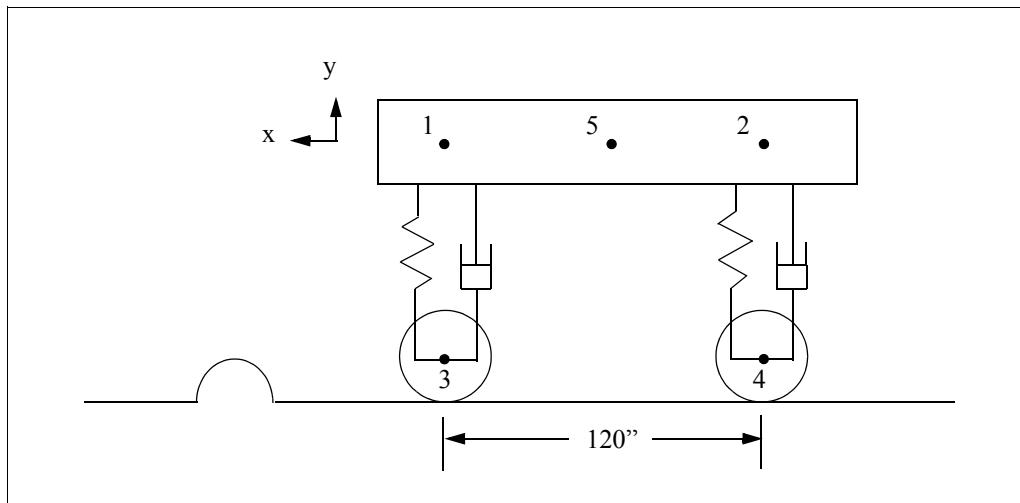


Figure 8-33 Acoustic Optimization Sound Pressure Levels: Initial and Final Distributions



RMS Response

This example simulates a car subjected to random loading that is fully correlated. The goal of this analysis is to minimize the RMS displacement at the center of gravity (cg) of the vehicle due to the random loading. A sketch of the simple vehicle representation is given below.



The goal of the optimization task is to minimize the RMS displacement of the vehicle CG while constraining the stress. Design variables are the stiffness and damping (spring and shock absorber) of the suspension. Numerical values for the design limits are:

$$100 < K < 1000$$

$$1.0 < B < 10.$$

$$|\sigma| \leq 4500$$

The complete input data file is given in [Listing 8-23](#). Two subcases are required in this case -- one for the front wheel and one for the back wheel. A unit enforced displacement is applied to each wheel using the enforced motion method.

Listing 8-23 Input Data File for DSOUG11

```

$      DSOUG11.DAT      - RANDOM INPUT AT WHEELS - ENFORCED MOTION
$      DIRECT METHOD
$      STRESS CONSTRAINTS AT DISCRETE FREQ
$      MINIMIZE ACCE AT GRID PT 5
$      CROSS SPECTRUM
$      USING NEW ENFORCED MOTION
$      ID MSC, DSOUG11 $
$      SOL 200 $
$      CEND
$      TITLE = SIMPLE CAR WITH RANDOM INPUT

```



```

SPC = 100
FREQUENCY = 130
STRESS (PHASE) = ALL
DISP(PHASE) = ALL
DESOBJ= 101
ANALYSIS = DFREQ
DESSUB = 800
$
$ THIS RANDOM CASE CONTROL CALLOUT IS NEEDED ONLY
$ FOR THE XYPILOT REQUEST WHEN USING OPTIMIZATION
$
RANDOM = 1000
SUBCASE 1
    DLOAD = 111
SUBCASE 2
    DLOAD = 112
$
OUTPUT (XYPILOT)
XTITLE = FREQUENCY (HZ)
YTITLE = DISP PSD AT GRID PT 5
XYPUNCH DISP PSDF /5(T2)
$
BEGIN BULK
$
$ CAR BODY
$
GRID 1 0. 0. 0.
GRID 2 120. 0. 0.
GRID 5 60. 0. 0.
$
$ WHEELS
$
GRID 3 0. -10. 0.
GRID 4 120. -10. 0.
$
CBAR 5 11 1 5 0. 1. 0.
CBAR 6 11 5 2 0. 1. 0.
$
PBAR 11 12 1.E2 1.E3 1.E3
    100. 0. -100. 0.
MAT1 12 3.E+7 7.8E-4
$
$ CONSTRAIN TO A PLANAR PROBLEM
$
SPC1 100 1345 1 2 5
SPC1 100 13456 3 4
SPC1,100,2,3,4
$
CONNM2 10 1 2.5
CONNM2 15 2 2.5
CONNM2 20 5 5.0
CONNM2 25 3 2.5
CONNM2 26 4 2.5
$
CBUSH 100 1000 1 3 0
CBUSH 200 1000 2 4 0
PBUSH 1000 K 200. 0
    B 2.
$
RLOAD2 111 222 444 DISP
SPCD 222 3 2 1.0
TABLED1 444
    0. 1. 100. 1. ENDT
$
RLOAD2 112 223 444 DISP
SPCD 223 3 2 1.0
$
FREQ2 130 0.1 2. 20

```



```

$  

$ DEFINE THE INPUT PSD  

$  

RANDPS 1000 1 1 1. 0. 145  

RANDPS 1000 2 2 1. 0. 145  

RANDPS 1000 1 2 1. 0. 146  

RANDPS 1000 1 2 0. 1. 147  

TABRND1 145  

.1 .1 5. 1. 10. .05 ENDT  

$  

TABRND1 146  

0.1000 0.0990 0.1096 0.1005 0.1202 0.1022 0.1318 0.1040  

0.1445 0.1059 0.1585 0.1079 0.1738 0.1101 0.1905 0.1124  

0.2089 0.1147 0.2291 0.1172 0.2512 0.1197 0.2754 0.1221  

0.3020 0.1246 0.3311 0.1269 0.3631 0.1289 0.3981 0.1305  

0.4365 0.1314 0.4786 0.1315 0.5248 0.1304 0.5754 0.1277  

0.6310 0.1228 0.6918 0.1151 0.7586 0.1038 0.8318 0.0879  

0.9120 0.0665 1.0000 0.0382 1.0965 0.0020 1.2023 -0.0434  

1.3183 -0.0986 1.4454 -0.1636 1.5849 -0.2372 1.7378 -0.3159  

1.9055 -0.3934 2.0893 -0.4593 2.2909 -0.4984 2.5119 -0.4910  

2.7542 -0.4149 3.0200 -0.2508 3.3113 0.0072 3.6308 0.3364  

3.9811 0.6685 4.3652 0.8819 4.7863 0.8223 5.2481 0.3435  

5.7544 0.2965 6.3096 0.6842 6.9183 0.5743 7.5858 0.0891  

8.3176 0.2820 9.1201 0.1964 10.0000 0.0063 ENDT  

$  

TABRND1 147  

0.1000 0.0142 0.1096 0.0159 0.1202 0.0177 0.1318 0.0198  

0.1445 0.0221 0.1585 0.0248 0.1738 0.0279 0.1905 0.0313  

0.2089 0.0352 0.2291 0.0397 0.2512 0.0448 0.2754 0.0506  

0.3020 0.0572 0.3311 0.0648 0.3631 0.0734 0.3981 0.0832  

0.4365 0.0944 0.4786 0.1070 0.5248 0.1212 0.5754 0.1371  

0.6310 0.1547 0.6918 0.1741 0.7586 0.1951 0.8318 0.2173  

0.9120 0.2401 1.0000 0.2625 1.0965 0.2830 1.2023 0.2993  

1.3183 0.3084 1.4454 0.3061 1.5849 0.2875 1.7378 0.2467  

1.9055 0.1775 2.0893 0.0749 2.2909 -0.0631 2.5119 -0.2318  

2.7542 -0.4160 3.0200 -0.5848 3.3113 -0.6898 3.6308 -0.6687  

3.9811 -0.4624 4.3652 -0.0505 4.7863 0.4968 5.2481 0.8888  

5.7544 0.8037 6.3096 0.3101 6.9183 -0.2722 7.5858 -0.5008  

8.3176 -0.2390 9.1201 0.0928 10.0000 0.0496 ENDT  

$  

$-----  

$ DESIGN MODEL  

$  

DESVAR,1,K2,1.,0.5,5.  

DESVAR,2,B2,1.,0.5,5.  

$  

DVPREL1,101,PBUSH,1000,K2,100.  

,1,200.  

DVPREL1,201,PBUSH,1000,B2,1.  

,2,2,0  

$  

DRESP1,101,MING5T2,RMSDISP,,,2,1000,5  

$  

DCONSTR,800,801,,4500.  

DCONSTR,800,802,,4500.  

DRESP1,801,E5PTC,FRSTRE,PBAR,,12,,11  

DRESP1,802,E5PTE,FRSTRE,PBAR,,14,,11  

$  

DOPTPRM DESMAX 20 P1 1 P2 14 METHOD 3$  

ENDDATA

```

The stress constraints are applied to the stresses resulting from these unit displacements in the individual subcases while the RMS response is calculated by integrating the results from the two subcases as given by [Equation \(2-19\)](#) on [page 44](#).



Unlike a non-optimization run, where the RANDPS is called out by the RANDOM Case Control command, the optimization run references the RANDPS entries through the ATTB field (field 8) on the DRESP1 entry. There are four RANDPS entries for this case. The first and second RANDPS entries represent the same autospectrum applied at the front and back wheel, respectively. The third and fourth RANDPS entries represent the real and imaginary parts of the cross spectrum between the front and back wheels. Note that these spectra are applied the same way as in a non-optimization run. The RTYPE field (field 4) on the DRESP1 entry specifies that an RMS displacement is to be used by specifying RMSDISP. In this example, we use the rms displacement (RMSDISP) as the objective function.

[Listing 8-24](#) shows the design history for this example while [Figure 8-34](#) shows the objective as a function of design cycle and [Figure 8-35](#) shows the design variable values. We see that the damping variable goes to its upper bound while the stiffness is reduced relative to its initial value. [Figure 8-36](#) shows the initial and final PSD plots . The figures also show that it is the area under these two curves that is reduced as the objective RMS displacement decrease from 1.906 to 0.627 inches.

Listing 8-24 Design History

```
*****
SUMMARY OF DESIGN CYCLE HISTORY
*****
(HARD CONVERGENCE ACHIEVED)
(SOFT CONVERGENCE ACHIEVED)
NUMBER OF FINITE ELEMENT ANALYSES COMPLETED 16
NUMBER OF OPTIMIZATIONS W.R.T. APPROXIMATE MODELS 15
OBJECTIVE AND MAXIMUM CONSTRAINT HISTORY
-----
          OBJECTIVE FROM      OBJECTIVE FROM      FRACTIONAL ERROR      MAXIMUM VALUE
CYCLE      APPROXIMATE      EXACT            OF                  OF
NUMBER     OPTIMIZATION     ANALYSIS        APPROXIMATION      CONSTRAINT
-----
INITIAL           1.905990E+00           -1.686030E-01
1       3.167362E-01   2.042072E+00   -8.448946E-01   1.188775E-01
2       -2.901167E+00  1.978038E+00  -2.466689E+00  -1.239245E-01
3       1.654747E+00   1.460800E+00  1.327672E-01  -2.961552E-01
4       -3.195068E-01  1.519903E+00  -1.210215E+00  -3.626885E-01
5       8.443491E-01   1.115222E+00  -2.428867E-01  N/A
6       8.663468E-01   1.082970E+00  -2.000269E-01  N/A
7       1.030138E+00   1.035083E+00  -4.777319E-03  N/A
8       8.189175E-01   8.873507E-01   -7.712077E-02  N/A
9       7.427580E-01   8.166540E-01   -9.048622E-02  N/A
10      7.328084E-01   7.472574E-01   -1.933596E-02  N/A
11      7.117122E-01   7.064722E-01   7.417253E-03  N/A
12      6.858875E-01   6.758027E-01   1.492262E-02  N/A
13      6.570486E-01   6.473593E-01   1.496738E-02  N/A
14      6.325558E-01   6.244484E-01   1.298324E-02  N/A
1      SIMPLE CAR WITH RANDOM INPUT                         SEPTEMBER 8, 2009 MSC NASTRAN 9/ 4/09 PAGE 151
0
-----
          OBJECTIVE FROM      OBJECTIVE FROM      FRACTIONAL ERROR      MAXIMUM VALUE
CYCLE      APPROXIMATE      EXACT            OF                  OF
NUMBER     OPTIMIZATION     ANALYSIS        APPROXIMATION      CONSTRAINT
-----
15       6.244484E-01   6.244484E-01   0.000000E+00  N/A
```



```

-----
1 SIMPLE CAR WITH RANDOM INPUT                               SEPTEMBER 8, 2009 MSC NASTRAN 9/ 4/09 PAGE 152
0

DESIGN VARIABLE HISTORY
-----
INTERNAL | EXTERNAL |          |
DV. ID. | DV. ID. | LABEL | INITIAL : 1 : 2 : 3 : 4 : 5 :
-----
1 | 1 | K2 | 1.0000E+00 : 1.1999E+00 : 9.6775E-01 : 1.1583E+00 : 9.2577E-01 : 8.3549E-01 :
2 | 2 | B2 | 1.0000E+00 : 1.1981E+00 : 1.4300E+00 : 1.7116E+00 : 2.0532E+00 : 2.2553E+00 :

INTERNAL | EXTERNAL |          |
DV. ID. | DV. ID. | LABEL | 6 : 7 : 8 : 9 : 10 : 11 :
-----
1 | 1 | K2 | 7.5635E-01 : 6.8345E-01 : 6.2430E-01 : 5.6517E-01 : 5.0948E-01 : 5.0075E-01 :
2 | 2 | B2 | 2.4689E+00 : 2.7028E+00 : 2.9366E+00 : 3.2148E+00 : 3.5331E+00 : 3.8586E+00 :

INTERNAL | EXTERNAL |          |
DV. ID. | DV. ID. | LABEL | 12 : 13 : 14 : 15 : 16 : 17 :
-----
1 | 1 | K2 | 5.0075E-01 : 5.0075E-01 : 5.0075E-01 : 5.0075E-01 :
2 | 2 | B2 | 4.2241E+00 : 4.6242E+00 : 5.0000E+00 : 5.0000E+00 :

*** USER INFORMATION MESSAGE 6464 (DOM12E)
RUN TERMINATED DUE TO HARD CONVERGENCE TO AN OPTIMUM AT CYCLE NUMBER =      15.
1 SIMPLE CAR WITH RANDOM INPUT                               SEPTEMBER 8, 2009 MSC NASTRAN 9/ 4/09 PAGE 153

```

————— OBJECTIVE FUNCTION--PSDF RMS DISP AT GRID PT5

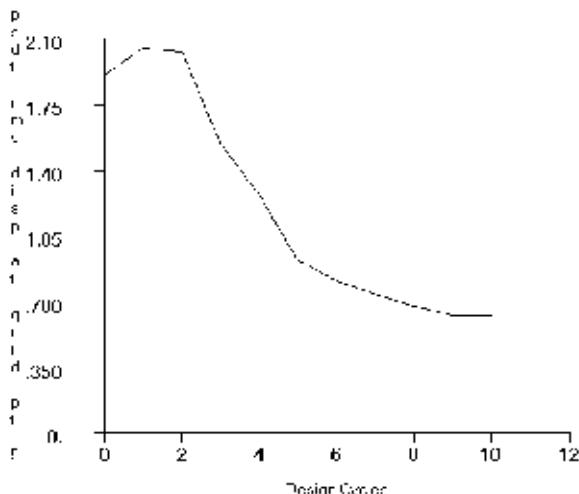


Figure 8-34 Objective Function

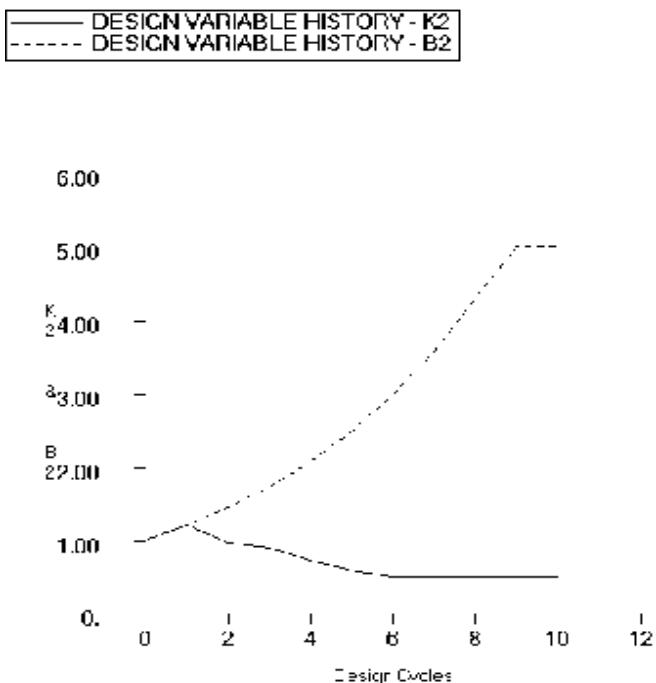


Figure 8-35 Design Variables



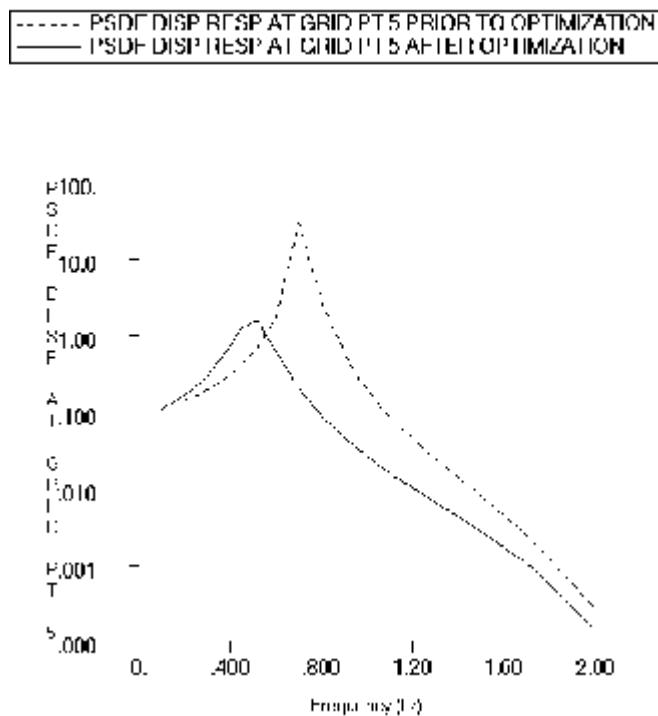


Figure 8-36 Comparison of PSDF DISP



Transient Dynamic Optimization

This simple example demonstrates the ability to perform structural optimization in conjunction with a transient analysis. The example is a portal frame modeled with three bar elements that is excited by an enforced motion at its base. This is a simple representation of a civil engineering structure subjected to a seismic load.

The frame is subjected to a horizontal ground motion in the form of a half sine pulse:

$$u_e(t) = 1.0 \sin(30t) \quad 0.005 \leq t \leq \pi/30 + 0.005$$

The standard enforced displacement techniques that are used in SOL's 109 and 112 are not available in SOL 200. Instead, this example employs the large mass method and offsets the start time of the enforced displacement to avoid having to specify initial conditions, which are also not supported in SOL 200.

The design task is to minimize the weight of the structure while imposing limits on the rms structural displacements and stresses as well as on the first natural frequency:

$$f_i > 9.55 \text{ Hz.}$$

$$\text{RMS Displacement} \leq 0.1 \text{ in.}$$

$$\text{RMS Stress} \leq 3000 \text{ psi}$$

The frame is modeled using three finite element bars (symmetry of the problem could have been used to limit this to two elements). This example has been adapted from a PhD thesis (see Cassis, J.H., in Reference 15.) and has some non-standard modeling techniques that are included here because they illustrate the use of DVPREL2 and DRESP2 entries. In particular, the two design variables for the problem are the bending inertia of the horizontal and vertical beams. The area of the beams is computed using:

$$A = 0.465 \sqrt{I}$$

and the stress at the end of the bars is computed as the ratio of the bending moment divided by the section modulus, where the section modulus is given by:

$$S = \sqrt{60.6I + 84000} - 290$$

The transient response was performed over a period of 250 milliseconds and responses were retained at every 5 milliseconds. An RMS response is computed by using the RSS option for the ATTB field on the DRESP1 entries for transient displacement and transient force response and then dividing the result by 51.0 (the number of samples used to compute the RSS).

The input file for this example is shown in [Listing 8-25](#). It is seen that there are two subcases, with the first subcase performing a modal analysis while the second performs the modal transient analysis (since the same METHOD entry is used for both subcases and the boundary condition does not change, only one modal analysis is performed). The enforced displacement is applied using a time history provided on a TLOAD2 entry that also points to a DAREA entry that provides the magnitude of the displacement.



Listing 8-25 Input Data File for DSOUG12

```

ID MSC, DSOUG12 $ V2004 EHJ 25-JUN-2003
$ Modified 2-Apr-2009 ehj MDR$
$TIME 10
TIME 60
SOL 200
CEND
TITLE= TRANSIENT DYNAMIC OPTIMIZATION - DSOUG12
METHOD = 300
DISPL = ALL
DESOBJ = 1000
DSAPRT(BY=1)
SUBCASE 2
DESSUB = 2
ANALYSIS = MODES
SUBCASE 3
DESSUB = 3
STRESS = ALL
FORCE = ALL
DLOAD = 310
TSTEP = 335
ANALYSIS = MTRAN
BEGIN BULK
$ EIGENVALUE EXTRACTION METHOD
EIGRL 300          10
$
$-----
$ ANALYSIS MODEL
$-----
$ 
GRID   1           0.    0.    0.    23456
GRID   2           0.    180.   0.    345
GRID   3           180.   180.   0.    345
GRID   4           180.   0.    0.    23456
$ VERTICAL COLUMNS
CBAR   1    101    1    2    -1.    0.    0.
CBAR   2    101    4    3    -1.    0.    0.
$ HORIZONTAL BAR
CBAR   3    102    2    3    0.    1.    0.
$
MAT1   200    30.E6      0.33    0.28
PARAM  WTMASS  2.588E-3
PARAM  POST    -1
$
PBAR   101    200    7.92    290.      10.
PBAR   102    200    7.92    290.      10.
$
RBE2   10     1     1     4
SUPORT 1         1
$ TRANSIENT LOADING DATA
$...SPECIFY HALF-SINE WAVE ENFORCED DISPLACEMENT BEGINNING AT T=.005
$  SPECIFY TIME DELAY SO AS NOT TO REQUIRE INITIAL CONDITIONS
TLOAD2 310    320      1     .005    .10972  4.77465 -90.
        0.0    0.0
$...TIME STEPPING INFORMATION:
TSTEP   335    250     0.001    5
$ LARGE MASS METHOD FOR SPECIFYING TRANSIENT ENFORCED DISPLACEMENTS
CMASS2 11     2.0E9   1     1
DAREA   320     1     1     5.176E6
$-----

```



```

$ DESIGN MODEL
DESVAR 1      X1      1.0      .1      10.
DESVAR 2      X2      1.0      0.1     10.
$
$...DV TO I RELATIONS:
DVPREL1 10    PBAR    101    I1      1.0      0.0
              1      1200.
DVPREL1 11    PBAR    102    I1      1.0      0.0
              2      1200.
$
$...DV TO A RELATIONS:
DVPREL2 12    PBAR    101    A       0.1      200
              DESVAR 1
DVPREL2 13    PBAR    102    A       0.1      200
              DESVAR 2
DEQATN 200    AREA(X) = .465 * SQRT(1200.*X)
$
DOPTPRM DESMAX 20      IPRINT 7      P1      1      P2      15
$...DEFINE WEIGHT OBJECTIVE:
DRESP1 1      W       WEIGHT
DRESP2 1000   SWEIG   1000
              DRESP1 1
DEQATN 1000   SWEIG(WEIGHT) = WEIGHT/100.
$
$...FREQUENCY CONSTRAINT:
DRESP1 2      L1      FREQ          2
DCONSTR 2     2      9.55
$
$...TRANSIENT DISPLACEMENT CONSTRAINTS:
DRESP1 3      UX2      TDISP          1      RSS      2
DRESP2 30     RMS      30
              DRESP2 3
DEQATN 30     RMS(RSS) = RSS / 51.
DCONSTR 3     30      0.10
$
$...TRANSIENT BENDING STRESS CONSTRAINTS:
$.....VERTICAL COLUMN, STRESSES AT ENDS A AND B:
DRESP1 5      M1A1    TFORC   ELEM      2      RSS      1
DRESP1 6      M1B1    TFORC   ELEM      4      RSS      1
DRESP2 10     SIGA1   210
              DESVAR 1
              DRESP2 5
DRESP2 11     SIGB1   210
              DESVAR 1
              DRESP2 6
$....HORIZONTAL BEAM, STRESS AT END A
DRESP1 7      M2A1    TFORC   PBAR      2      RSS      102
DRESP2 12     SIGA2   210
              DESVAR 2
              DRESP2 7
DEQATN 210    BSIG(X,M) = M/( SQRT(72720. *X + 84000.) - 290. )/ 51.
DSCREEN EQUA  -10.
DSCREEN DISP  -10.
DCONSTR 3     10      +3.E3
DCONSTR 3     11      +3.E3
DCONSTR 3     12      +3.E3
ENDDATA

```

The initial values of the two DESVAR's are set to 1.0 with the beam bending inertias being set to 1200. times the design variable value. The inertia values are linear in the design variable and can therefore be input on a DVPREL1 entry. The area, however, is nonlinear in the design variable and this requires



DVPREL2 entries. The responses are standard, but the RMS responses can perhaps benefit from a description. DRESP1 3 selects a TDISP (transient displacement) response at GRID 2, component 1. The RSS in the ATTB field for this entry indicates that the responses that are computed from this DRESP1 are to be converted into a RSS value. The output of this process is scaled by 51., reflecting the 51 time steps that went into the RSS calculation. Note that the DRESP2 that is used to provide this RMS responses invokes the RSS values using a DRESP2 label on its continuation even though the RSS value was created using a DRESP1. This is a special rule for type-1 responses that use the character inputs in the ATTB field: subsequent references to the response must identify it as a DRESP2. The limit of 0.1 in. on the RMS displacement is imposed as an upper bound on the DRESP2 value. Three DRESP1 entries create RMS values for the bending moments at both ends of the one of the vertical beams and at the left end of the horizontal beam (symmetry considerations make it unnecessary to impose constraints on the left side of the structure). This value is converted into an RMS stress value by dividing by 51.0.

Selected output is shown for this job in [Listing 8-26](#). Attention is directed to the fragment that prints the final response values. The RSS response type responses create some special results that are explained here. The output fragment begins with prints of the final response values that are produced by the DRESP1 entries. Data are shown for only two of the time steps, but the actual output file contains these data for all 51 time steps. This is followed by a print of the RETAINED DRESP2 RESPONSES. It is seen that the first four DRESP2 responses have the same response label as the DRESP1 values and that the Equation ID for all of them is RSS. These responses are the RSS values that are computed from the DRESP1 transient responses. This is followed by the final values for four actual DRESP2 responses that are constrained in the design task. It is seen that two of the stress responses and the displacement response are very close to their limit values.

The design history shows that a converged design is obtained in 6 cycles. The initial weight is 7835. pounds (this includes 5400. pounds of non-structural mass) while the final weight is 7404.

----- MODAL TRANSIENT FORCE RESPONSES -----								
INTERNAL ID	DRESP1 ID	RESPONSE LABEL	ELEMENT ID	COMPONENT NO.	TIME	LOWER BOUND	VALUE	UPPER BOUND
202	5	M1A1		1	2	2.4500E-01	N/A	-2.6437E+06
203	6	M1B1		1	4	2.4500E-01	N/A	1.6927E+06
204	7	M2A1		3	2	2.4500E-01	N/A	1.6927E+06
----- MODAL TRANSIENT FORCE RESPONSES -----								
INTERNAL ID	DRESP1 ID	RESPONSE LABEL	ELEMENT ID	COMPONENT NO.	TIME	LOWER BOUND	VALUE	UPPER BOUND
206	5	M1A1		1	2	2.5000E-01	N/A	-2.5950E+06
207	6	M1B1		1	4	2.5000E-01	N/A	1.6619E+06
208	7	M2A1		3	2	2.5000E-01	N/A	1.6619E+06
----- RETAINED DRESP2 RESPONSES -----								
INTERNAL ID	DRESP2 ID	RESPONSE LABEL	EQUATION ID	LOWER BOUND	VALUE	UPPER BOUND		
1	1000	SWEIG	1000	N/A	7.4040E+01	N/A		
2	3	UX2	RSS	N/A	5.1134E+00	N/A		
3	5	M1A1	RSS	N/A	1.2452E+07	N/A		
4	6	M1B1	RSS	N/A	7.9735E+06	N/A		
5	7	M2A1	RSS	N/A	7.9735E+06	N/A		
6	10	SIGA1	210	-1.0000E+20	2.8445E+03	3.0000E+03		
7	11	SIGB1	210	-1.0000E+20	1.8214E+03	3.0000E+03		
8	12	SIGA2	210	-1.0000E+20	2.8438E+03	3.0000E+03		
9	30	RMS	30	-1.0000E+20	1.0026E-01	1.0000E-01		





External Response to Include Alternative Buckling Response

This example demonstrates how the DRESP3 Bulk Data entry can be applied to include a buckling criteria that is not available directly from MSC Nastran. The application is to a bar loaded longitudinally as shown in [Figure 8-37](#). This particular design example is so simple that it could be solved using hand calculations. However, the contribution of this example is not in the complexity of the problem solved but in its illustration of the process you must follow to apply the external response capability.

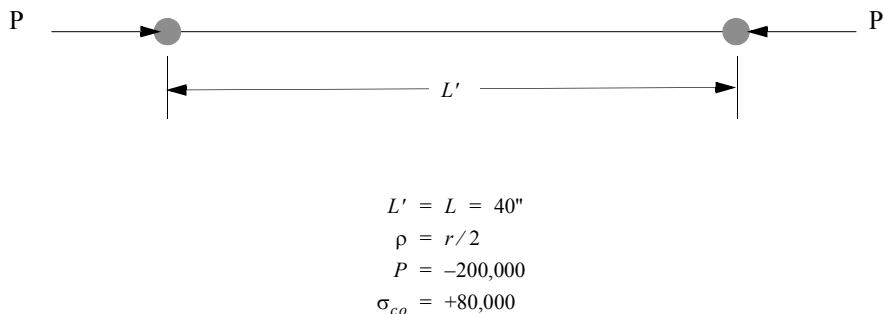


Figure 8-37 Pinned-Pinned Column Buckling

The design task is to find the area of the rod that minimizes the weight while satisfying constraints on the longitudinal stress and on the buckling behavior. The stress constraint is

$$-80,000 \leq \sigma \leq 100,000$$

while the critical buckling stress is computed using a combination of Johnson and Euler buckling criteria, with the selection of the criterion to use based on the slenderness ratio of the rod (see Peery and Azar in Reference 16.).

$$\sigma_{cr} = \left[1 - \frac{\sigma_{co}(L'/\rho)^2}{4\pi^2 E} \right] \sigma_{co} \quad L'/\rho \leq \pi \sqrt{\frac{2E}{\sigma_{co}}} \quad (8-17)$$

$$\sigma_{cr} = \frac{\pi^2 E}{(L'/\rho)^2} \quad L'/\rho \geq \pi \sqrt{\frac{2E}{\sigma_{co}}} \quad (8-18)$$

σ_{cr} = critical buckling stress

L' = effective length

ρ = radius of gyration

σ_{co} = empirically determined column yield stress



The fact that the failure criteria can branch based on a combination of structural properties and responses makes it an ideal application for the external response capability. Listing 8-26 shows the input file for this example. It is seen that there are two subcases with the first subcase performing a statics analysis and imposing constraints on the stress in the element and on the combined buckling failure criteria. The second subcase performs a buckling analysis and places a lower bound of 1.0 on the first buckling eigenvalue. This is redundant with the Euler buckling condition of the first subcase and therefore provides a check on the external response calculation.

Listing 8-26 Input Data File for the DRESP3 Example (DSOUG13)

```

CONNECT DRESP3 TESTGRP EXTRESP
$ID MSC, DSOUG13 $ v2004 ehj 25-Jun-2003
DIAG 8,15 $
SOL 200
CEND
TITLE      = BUCKLING TEST CASE WITH AN EXTERNAL RESPONSE - DSOUG13
SUBTITLE   = JOHNSON/EULER BUCKLING CASE
ECHO       = SORT
SPC        = 100
DESOBJ    = 20
SUBCASE 1
DESSUB    = 1
LABEL     = LOAD CONDITION 1
LOAD      = 300
ANALYSIS  = STATICS
DISP      = ALL
STRESS    = ALL
SUBCASE 2
DESSUB    = 2
ANALYSIS  = BUCK
METHOD    = 1
DISP      = ALL
LABEL     = BUCKLING FACTORS
$
BEGIN BULK
PARAM,POST,-1
$-----
$ ANALYSIS MODEL
$-----
$ 
$ GRID DATA
$   2      3      4      5      6      7      8      9      10
GRDSET          345
GRID  1           0.0     0.0     0.0
GRID  2           8.0     0.0     0.0
GRID  3          16.0     0.0     0.0
GRID  4          24.0     0.0     0.0
GRID  5          32.0     0.0     0.0
GRID  6          40.0     0.0     0.0
GRID 10          0.0     0.0   100.0      123456
$
$ ELEMENT AND MATERIAL DATA
CBAR   1      10      1      2      10
CBAR   2      10      2      3      10
CBAR   3      10      3      4      10
CBAR   4      10      4      5      10
CBAR   5      10      5      6      10
MAT1   1      3.0E7     0.33    0.1
$ PROPERTY DATA

```



```

PBARL 10      1      MSCBML0ROD
      1.0
$ BOUNDARY CONDITION DATA
SPC1 100      1      1
SPC1 100      2      1      6
$ EXTERNAL LOADS DATA
FORCE 300      6      -2.0E5  1.0
$ BUCKLING ANALYSIS DATA
EIGRL 1      .05      4
$
$-----
$ DESIGN MODEL
$-----
$DESVAR 1      RG 1.0      0.01      10.0
DOPTPRM P215   DESMAX 20      DELP 0.5      GMAX 0.01
      CONVDV 0.01      CONVPR 0.02      P1      1
DRESP1 20      W      WEIGHT
DRESP1 23      S1      STRESS  PBAR      7      10
DRESP1 24      S1      STRESS  PBAR      8      10
DRESP1 25      S1      STRESS  PBAR      6      10
DRESP3 32      JOHNSON  TESTGRP  EULJOH
      DESVAR 1
      DTABLE L      E      SIGMAC
      DRESP1 25
$
DCONSTR 1      23      100000.
DCONSTR 1      24      -80000.
DCONSTR 1      32      1.0
DTABLE E      30.0E6  L      40.0      SIGMAC  8.0E4
DVPREL1 10     PBARL 10      DIM1
      1      1.0
$
$ DESIGN FOR BUCKLING EIGENVALUE
DRESP1 1      BUCK1  LAMA      1
DCONSTR 2      1      1.0
PARAM  DSNOKD  1.0
ENDDATA

```

The listing shows the DRESP3 entry that provides the buckling response. It is seen that the inputs include the design variable value, constants that define E, the rod length and σ_{co} , and the axial stress in the rod. TESTGRP is specified on the DRESP3 entry as the external response GROUP and EULJOH as the TYPE. A CONNECT file management statement specifies a DRESP3 application, TESTGRP as the group, and EXTRESP as the external evaluator.

Two subroutines are modified to perform the server task. As explained in [DRESP3 Bulk Data Entries](#), the entire contents of the `/MSC2012/dr3srv/` directory need to be copied to your local directory. The two subroutines that require modification are members of that directory. The first is entitled R3SGRT and is shown in [Listing 8-27](#). R3SGRT is used to specify the types of responses that are available from the server. This is used to make sure that the user input TYPE is supported by the server. In this case, there is only one type of response supported and it is EULJOH. The second subroutine is entitled R3SVALD (shown in [Listing 8-29](#)) and performs the response evaluation based on the user input arguments. (Note that the D suffix on this subroutine name is used when running on a short word; i.e., 32 bit, machine and that the subroutine named R3SVALS should be modified when a long word machine is used). In this case, there are four real arguments being passed and their order is given on the DRESP3 entry. The output of



the subroutine is the DR3VAL argument and is calculated using the formulas given in [Equation \(8-17\)](#) and [Equation \(8-18\)](#).



Listing 8-27 The R3SGRT Subroutine

```

SUBROUTINE R3SGRT(GRPID,TYPNAM,nresp, grdtyp, ERROR)
C -----
C
C PURPOSE: VERIFY THE EXTERNAL RESPONSE TYPE
C
C GRPID   input integer      - Group ID
C TYPNAM  input character*8   - Name of external response type
C nresp    output integer     - number of responses for this dresp3
C grdtyp   output integer     - integer array of length nresp
C                               indicating how gradients are to be
C                               computed
C                               = 1 user to supply analytic gradients
C                               varies during approx. optimization
C                               = 2 user to supply analytic gradients
C                               invariant during approx. optimization
C                               = 3 finite difference technique to provide gradients
C                               varies during approx. optimization
C                               = 4 finite difference technique to provide gradients
C                               invariant during approx. optimization
C
C ERROR   input/output integer -error code for the call.
C
C Method
C Match the user input: typnam with the list of available
C external response types. Once a match is found, nresp and grdtyp
C are set. If no match is found, set error code.
C
C Called by
C          R3CGRT
C
C NOTE:
C The writer of this routine is responsible to specify
C NTYPES and R3TYPE.
C -----
C
C VARIABLES PASSED IN/out
C
C INTEGER GRPID, ERROR, nresp
C integer grdtyp(*)
C CHARACTER*8 TYPNAM
C
C LOCAL VARIABLES
C
C INTEGER NTYPES, BADTYP
C PARAMETER(NTYPES=6)
C CHARACTER*8 R3TYPE(NTYPES)
C
C DATA BADTYP/7554/
C DATA R3TYPE/'USEVAR1 ','USEVAR10','USEALL',
C .           'USEMIXVS','FREQMOD ','EULJ0H ',/
C
C ERROR = 0
C DO 100 ITYPE = 1, NTYPES
C       IF (TYPNAM .EQ. R3TYPE(ITYPE)) THEN
C           nresp = 1
C           grdtyp(1) = 3
C           GOTO 200
C       END IF
100    CONTINUE
C       ERROR = BADTYP
200    CONTINUE
C
C RETURN
C END

```



Listing 8-28 The R3SVALD Subroutine

```

SUBROUTINE R3SVALD(GRPID,TYPNAM,
.          NITEMS,ARGLIS,
.          NSIZE, ARGVAL,
.          NWRDA8,ARGCHR,
.          forg,nresp,narg,
.          DR3VAL,senval, error )
C -----
C
C PURPOSE: COMPUTE THE EXTERNAL RESPONSE
C
C GRPID   INPUT INTEGER      - GROUP ID
C TYPNAM  INPUT CHARACTER*8   - NAME OF EXTERNAL RESPONSE TYPE
C NITEMS   INPUT INTEGER     - DIMENSION OF ARRAY ARGLIS (=12)
C NSIZE    INPUT INTEGER     - DIMENSION OF ARRAY ARGVAL
C NWRDA8   INPUT INTEGER     - DIMENSION OF CHARACTER ARRAY ARGCHR
C ARGLIS   INPUT INTEGER     - ARRAY OF NO. OF ITEMS FOR EACH
C                               ARGUMENT type
C ARGVAL   INPUT DOUBLE      - ARRAY OF ARGUMENT VALUES (except
c                               characters)
C ARGCHR   INPUT CHARACTER*8 - ARRAY OF CHARACTER VALUES
c nresp   input integer      - number of responses
c forg    input integer      - type of call
c                               = 0 function evaluation
c                               = 1 sensitivity evaluation
c narg    input integer      - number of arguments needing gradients
C DR3VAL   OUTPUT DOUBLE     - VALUE OF THE EXTERNAL RESPONSES
c senval  output double     - matrix of the sensitivity of the IRth
c                               response to the IARGth argument
C ERROR    INPUT/OUTPUT INTEGER -ERROR CODE FOR THE CALL.
c                               0 = PRINT ERROR MESSAGES
c                               1 = DO NOT PRINT ERROR MESSAGES.
C
C METHOD
C A) SET UP VARIOUS PARAMETERS FROM THE ARGUMENT LIST
C B) if forg = 0 EVALUATE THE EXTERNAL RESPONSE BASED ON THE
C    GIVEN TYPNAM
C C) else if forg = 1 EVALUATE THE sensitivities of the external
C    responses to the arguments that can vary for
C    the given typnam
C D) RETURN BADTYP ERROR IF TYPNAM IS NOT MATCHED HERE.
C
C nsize - the number of arguments or values in a dresp3 entry
C
C NSIZE=NV+NC+NR+NNC+NDVP1+NDVP2+NDVC1+NDVC2+NDVM1+NDVM2+NRR2
C
C where:
C nv      = number of desvars    nc      = number of dtables
C nr      = number of drespls    nnc     = number of dnode pairs
C ndvp1   = number of dvprells   ndvp2   = number dvprel2s
C ndvc1   = number of dvcrells   ndvc2   = number dvcrel2s
C ndvm1   = number of dvmrells   ndvm2   = number dvmrel2s
C nrr2    = number of dresp2s
C
C narg = nsize - nc
C CALLED BY
C SENDR3SVALD
C -----
C
C VARIABLES PASSED IN
C
CHARACTER*8 TYPNAM, ARGCHR(NWRDA8)
integer forg , nresp
INTEGER GRPID, NITEMS, NSIZE, ARGLIS(NITEMS), ERROR, NWRDA8
DOUBLE PRECISION ARGVAL(NSIZE), DR3VAL(*), senval(nresp,*)
C
C LOCAL VARIABLES
C

```



```

INTEGER BADTYP
DOUBLE PRECISION PI, FAC, FACT, SLNDER
DOUBLE PRECISION R,L,E,SIGMA,SIGMAC, RGYRA
C
DATA BADTYP /7554/
C
PI = 3.14159
PI2 = PI * PI
C
THE USER-SUPPLIED EQUATION TO DEFINE THE EXTERNAL RESPONSE
SIGMA = DRESP1, R=DESVAR, L, E AND SIGMAC = DTABLE CONSTANTS
C
EULER : EULER= -SIGMA * (L/ RGYRA ) **2 / (PI**2 * E)
RGYRA = R / 2.0
C
JOHNSON: JOHNSON = -SIGMA / (SIGMAC * FACTOR )
FACTOR = 1. - SIGMAC * (L/RGYRA)**2 /(4 * PI**2 * E)
ERROR = 0
C
SET UP PARAMETERS FOR VARIOUS ARGUMENT ITEMS
C
IF (TYPNAM .EQ. 'EULJOH ') THEN
    R      = ARGVAL(1)
    L      = ARGVAL(2)
    E      = ARGVAL(3)
    SIGMAC = ARGVAL(4)
    SIGMA  = ARGVAL(5)
    RGYRA  = R / 2.0
    SLNDER = L / RGYRA
    FACT  = PI * SQRT(2.0D0 * E / SIGMAC)
    IF ( SLNDER .LE.FACT) THEN
        C
        JOHNSON CRITERION
        FAC   = 1.0D0 - SIGMAC * (SLNDER) ** 2 /(4.0D0 * PI2 * E )
        DR3VAL(1) = -SIGMA / (SIGMAC * FAC)
    ELSE
        C
        Euler CRITERION
        DR3VAL(1) = -SIGMA * SLNDER**2 / (PI2 * E)
    ENDIF
ELSE
    ERROR = BADTYP
ENDIF
C
RETURN
END

```

Once the two files have been modified, the next step is to build the server command using the command:

```
msc2012 ./dr3srv build
```

This will create the external response server with the name dr3serv.

A final file that is required is the one that indicates where the server program resides and, in this case, is named '*evalconnect*' and has the form:

```
EXTRESP,-,/usr/ehj/dsoug/dr3srv/dr3serv
```

where EXTRESP is the evaluator name given on the CONNECT file management statement and the '-' connect option in this case indicates the server exists on any network mounted computer and this is followed by the name of the file which contains the server.

Now that all the files have been assembled, the job is submitted using the gmconn option; for example:



```
nastran dsoug13 scr=yes gmconn=evalconnect
```

Selected results for this simple design task are presented in [Listing 8-29](#). The printout first gives results from the final analysis and it is seen that the single design variable that the user has specified to be the radius of the ROD input on the PBARL has generated nine designed properties for the bar cross section. The “RESPONSE IN THE DESIGN MODEL” output indicates that the stresses in the bars are well below their compression allowable and that the final global buckling response is 1.0863. Since this is above the 1.0 limit that has been placed on this response, we can infer that the Johnson criterion is the critical design constraint. This is confirmed by the fact that all of the retained DRESP3 responses are at their upper bound limit. The design history summary indicates it was necessary to increase the rod radius from 1.0 to 1.1072 to satisfy this design requirement and that this was accomplished in two design cycles.



Listing 8-29 Final Results in the Buckling Design Task (DSOUG13)

***** ANALYSIS RESULTS BASED ON THE FINAL DESIGN *****						
----- DESIGN OBJECTIVE -----						
INTERNAL RESPONSE ID	TYPE OF RESPONSE	LABEL	MINIMIZE OR MAXIMIZE	SUPERELEMENT ID	SUBCASE ID	VALUE
1	DRESP1	WEIGHT	MINIMIZE	0	0	1.5405E+01
----- DESIGN VARIABLES -----						
INTERNAL ID	DESVAR ID	LABEL	LOWER BOUND	VALUE	UPPER BOUND	
1	1	RG	1.0000E-02	1.1072E+00	1.0000E+01	
----- DESIGNED PROPERTIES -----						
PROPERTY TYPE	PROPERTY ID	PROPERTY NAME	TYPE OF PROPERTY	LOWER BOUND	VALUE	UPPER BOUND
PBARL	10	DIM1	DVPREL1	1.0000E-15	1.1072E+00	1.0000E+20
PBAR	10	C1	DVREL1	-1.0000E+35	1.1072E+00	1.0000E+20
PBAR	10	D2	DVREL1	-1.0000E+35	1.1072E+00	1.0000E+20
PBAR	10	E1	DVREL1	-1.0000E+35	-1.1072E+00	1.0000E+20
PBAR	10	F2	DVREL1	-1.0000E+35	-1.1072E+00	1.0000E+20
PBAR	10	A	SECPRO	1.0000E-15	3.8512E+00	1.0000E+20
PBAR	10	I1	SECPRO	1.0000E-15	1.1803E+00	1.0000E+20
PBAR	10	I2	SECPRO	1.0000E-15	1.1803E+00	1.0000E+20
PBAR	10	J	SECPRO	1.0000E-15	2.3605E+00	1.0000E+20
----- DESIGN CONSTRAINTS ON RESPONSES -----						
(MAXIMUM RESPONSE CONSTRAINTS MARKED WITH **)						
INTERNAL ID	DCONSTR ID	INTERNAL RESPONSE ID	RESPONSE TYPE	INTERNAL REGION ID	SUBCASE ID	VALUE
1	1	2	STRESS	LOWER	1	-3.5085E-01
2	1	3	STRESS	LOWER	1	-3.5085E-01
3	1	4	STRESS	LOWER	1	-3.5085E-01
4	1	5	STRESS	LOWER	1	-3.5085E-01
5	1	6	STRESS	LOWER	1	-3.5085E-01
6	1	1	EXTERNAL	UPPER	32	2.7899E-03**
7	1	2	EXTERNAL	UPPER	32	2.7899E-03**
8	1	3	EXTERNAL	UPPER	32	2.7899E-03**
9	1	4	EXTERNAL	UPPER	32	2.7899E-03**
10	1	5	EXTERNAL	UPPER	32	2.7899E-03**
11	2	12	LAMA	LOWER	0	-8.6330E-02



RESPONSES IN DESIGN MODEL								
(N/A - BOUND NOT ACTIVE OR AVAILABLE)								
----- WEIGHT RESPONSE -----								
<hr/>								
INTERNAL ID	DRESP1 ID	RESPONSE LABEL	ROW ID	COLUMN ID	LOWER BOUND	VALUE	UPPER BOUND	
1	20	W	3	3	N/A	1.5405E+01	N/A	
----- FINAL ANALYSIS SUBCASE = 1								
----- STRESS RESPONSES -----								
<hr/>								
INTERNAL ID	DRESP1 ID	RESPONSE LABEL	ELEMENT ID	VIEW ELM ID	COMPONENT NO.	LOWER BOUND	VALUE	UPPER BOUND
2	24	S1	1	8	-8.0000E+04	-5.1932E+04	N/A	
3	24	S1	2	8	-8.0000E+04	-5.1932E+04	N/A	
4	24	S1	3	8	-8.0000E+04	-5.1932E+04	N/A	
5	24	S1	4	8	-8.0000E+04	-5.1932E+04	N/A	
6	24	S1	5	8	-8.0000E+04	-5.1932E+04	N/A	
7	25	S1	1	6	N/A	-5.1932E+04	N/A	
8	25	S1	2	6	N/A	-5.1932E+04	N/A	
9	25	S1	3	6	N/A	-5.1932E+04	N/A	
10	25	S1	4	6	N/A	-5.1932E+04	N/A	
11	25	S1	5	6	N/A	-5.1932E+04	N/A	
----- FINAL ANALYSIS SUBCASE = 2								
----- BUCKLING LOAD RESPONSES -----								
<hr/>								
INTERNAL ID	DRESP1 ID	RESPONSE LABEL	MODE NO.	LOWER BOUND	VALUE	UPPER BOUND		
12	1	BUCK1	1	1.0000E+00	1.0863E+00	N/A		
----- RETAINED DRESP3 RESPONSES -----								
<hr/>								
INTERNAL ID	DRESP3 ID	RESPONSE LABEL	GROUP NAME	TYPE NAME	LOWER BOUND	VALUE	UPPER BOUND	
1	32	JOHNSON	TESTGRP	EULJOH	N/A	1.0028E+00	1.0000E+00	
2	32	JOHNSON	TESTGRP	EULJOH	N/A	1.0028E+00	1.0000E+00	
3	32	JOHNSON	TESTGRP	EULJOH	N/A	1.0028E+00	1.0000E+00	
4	32	JOHNSON	TESTGRP	EULJOH	N/A	1.0028E+00	1.0000E+00	
5	32	JOHNSON	TESTGRP	EULJOH	N/A	1.0028E+00	1.0000E+00	
SUBCASE 2								

0



```
*****
SUMMARY OF DESIGN CYCLE HISTORY
*****
```

(HARD CONVERGENCE ACHIEVED)

(SOFT CONVERGENCE ACHIEVED)

NUMBER OF FINITE ELEMENT ANALYSES COMPLETED	3
NUMBER OF OPTIMIZATIONS W.R.T. APPROXIMATE MODELS	2

OBJECTIVE AND MAXIMUM CONSTRAINT HISTORY

CYCLE NUMBER	OBJECTIVE FROM APPROXIMATE OPTIMIZATION	OBJECTIVE FROM EXACT ANALYSIS	FRACTIONAL ERROR OF APPROXIMATION	MAXIMUM VALUE OF CONSTRAINT
INITIAL		1.256637E+01		4.017631E-01
1	1.539844E+01	1.539775E+01	4.471774E-05	3.486395E-03
2	1.540468E+01	1.540468E+01	1.238162E-07	2.789855E-03

0

DESIGN VARIABLE HISTORY

INTERNAL DV. ID.	EXTERNAL DV. ID.	LABEL	INITIAL	:	1	:	2	:	3	:	4	:	5	:
1	1	RG	1.0000E+00	:	1.1069E+00	:	1.1072E+00	:						

*** USER INFORMATION MESSAGE 6464 (DOM12E)
RUN TERMINATED DUE TO HARD CONVERGENCE TO AN OPTIMUM AT CYCLE NUMBER = 2.



Continuing the Design Process in a Subsequent Job

This final example section provides some guidance on continuing a design task across multiple job submittals. The preferred situation would be that design information gathered on an initial run be available in a subsequent run with a minimal amount of recalculation. This is related to the powerful feature in MSC Nastran referred to as automatic restarts (see Automatic Restarts, (see [RESTART](#) (p. 97) in the *MSC Nastran Quick Reference Guide*). However, design optimization has specialized techniques in this area that differ from those available in the other Solution Sequences. These techniques require more user intervention so that it is felt valuable to provide this section with design optimization examples so that you can benefit from the available capabilities. At the same time, there is limited support for the standard Restart capability of MSC Nastran and this support is also described by example. This section therefore focuses on three separate scenarios:

1. Continuing a property optimization design task
2. Continuing a shape optimization design task
3. Restarting from a previous analysis

Continuing a Property Optimization Design Task

The basic principle to recognize when continuing an optimization task is that, in SOL 200, the design model data overrides the analysis model (shape optimization is a special case, as discussed in the next subsection). This feature can be used to our advantage to continue from a previously obtained design.

As a design cycle is performed, PARAMeters DESPCH and DESPCH1 can be used to save critical design information to the .pch file that is created during the run. By default, DESVAR and property results are written to the .pch file following the last design cycle. It is a simple matter then to delete the existing DESVAR entries and insert the new DESVAR entries from the punch file to continue the design process.

We can illustrate this process using the Three-Bar Truss example of DSOUG1. [Listing 8-1](#) on page 491 does not contain any punch parameters, with the result that the .pch file produced by this run contains only final DESVAR and PROD properties as shown in [Listing 8-30](#). Note that, by default, this output is given using the double field format so that adequate data precision is obtained. (Small field format can be obtained by using negative integers for the DESPCH1 parameter.) The DESVAR entries from the punch file are used in place of the original DESVAR entries to produce the data file shown in [Listing 8-31](#). It is not necessary (but does not hurt) to include the PROD entries since the DESVAR values will be used to override the original properties. Some other changes have been made in the file of [Listing 8-31](#): the DLINK entry has been removed so that all three rod areas are now being designed and the DOPTPRM entry has been changed to that the SQP algorithm (METHOD=3) is used for optimization and a direct approximation (APRCOD=1) is used for the approximate model calculations.

It is noted that the NEWBULK option of the ECHO command (generation of a new Bulk Data File) provides a streamlined way of creating a complete updated bulk data deck in the .pch file. This same exercise could be performed by adding case control and other commands to the bulk data file and making the DLINK, METHOD and APPCOD edits discussed above.



Listing 8-30 The Punch File Produced by DSOUG1 in Section 7.1

```

$ ****
$ *      CONTINUOUS DESIGN CYCLE NUMBER =       6   *
$ ****
$ $ UPDATED DESIGN MODEL DATA ENTRIES
$ DESVAR *          1A1           8.36519659E-01  1.00000001E-01+D  1V
*D 1V 1.00000000E+02 1.00000000E+00
DESVAR *          2A2           3.29410642E-01  1.00000001E-01+D  2V
*D 2V 1.00000000E+02 1.00000000E+00
DESVAR *          3A3           8.36519659E-01  1.00000001E-01+D  3V
*D 3V 1.00000000E+02 1.00000000E+00
$ $ UPDATED ANALYSIS MODEL DATA ENTRIES
$ PROD*          11            1  8.36519659E-01  0.00000000E+00*
* 0.00000000E+00 0.00000000E+00
PROD*          12            1  3.29410642E-01  0.00000000E+00*
* 0.00000000E+00 0.00000000E+00
PROD*          13            1  8.36519659E-01  0.00000000E+00*
* 0.00000000E+00 0.00000000E+00

```

Listing 8-31 Three Bar Truss Updated with File from Listing 8-30

```

ID MSC DSOUG14 $
TIME 10 $
SOL 200 $ OPTIMIZATION
CEND
TITLE = SYMMETRIC THREE BAR TRUSS DESIGN OPTIMIZATION - DSOUG14
SUBTITLE = CONTINUATION - 3 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
SPC1 100 123456 1      THRU  3
$ 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)
$ 
$     UPDATED DESIGN MODEL DATA ENTRIES
$ 
DESVAR *          1A1      8.36519659E-01  1.00000001E-01+D  1V
*D    1V  1.00000000E+02  1.00000000E+00
DESVAR *          2A2      3.29410642E-01  1.00000001E-01+D  2V
*D    2V  1.00000000E+02  1.00000000E+00
DESVAR *          3A3      8.36519659E-01  1.00000001E-01+D  3V
*D    3V  1.00000000E+02  1.00000000E+00
$ 
$...DEFINITION OF DESIGN VARIABLE TO ANALYSIS MODEL PARAMETER RELATIONS
$DVPREL1 ID      TYPE      PID      NAME      PMIN      PMAX      CO      +
$+    DVID1    COEF1    DVID2    COEF2    ...
DVPREL1 10      PROD      11      A
1      1.0
DVPREL1 20      PROD      12      A
2      1.0
DVPREL1 30      PROD      13      A
3      1.0
$ 
$...STRUCTURAL RESPONSE IDENTIFICATION
$DRESP1 ID      LABEL      RTYPE      PTYPE      REGION      ATTA      ATTB      ATT1      +
$+    ATT2    ...
DRESP1 20      W       WEIGHT
DRESP1 21      U4      DISP
DRESP1 23      S1      STRESS    PROD
12
2
11
$ 
$...CONSTRAINTS
$DCONSTR DCID      RID      LALLOW      UALLOW
DCONSTR 21      21      -0.20      0.20
DCONSTR 21      23      -15000.  20000.
$ 
$...OPTIMIZATION CONTROL:
$ 
DOPTPRM DESMAX  10      METHOD  3      APRCOD  1$ 
$.....2.....3.....4.....5.....6.....7.....8.....9.....0
ENDDATA

```

The design cycle history for the new file is shown in [Listing 8-32](#). It is seen that there is a small change in the design in this case in terms of the objective function and design variables. It is reassuring to see that the first and third design variables remain identical even though there is no longer an explicit linking between the two variables.



Listing 8-32 Design Cycle History for the Continued Design Task

```
*****
SUMMARY OF DESIGN CYCLE HISTORY
*****
(HARD CONVERGENCE ACHIEVED)

NUMBER OF FINITE ELEMENT ANALYSES COMPLETED      3
NUMBER OF OPTIMIZATIONS W.R.T. APPROXIMATE MODELS   2

OBJECTIVE AND MAXIMUM CONSTRAINT HISTORY
-----
CYCLE          OBJECTIVE FROM          OBJECTIVE FROM          FRACTIONAL ERROR      MAXIMUM VI
NUMBER        APPROXIMATE          EXACT                OF                   OF
OPTIMIZATION          ANALYSIS           APPROXIMATION
-----
INITIAL          2.695446E+00
1              2.698603E+00          2.698606E+00      -1.148534E-06      9.82421
2              2.698332E+00          2.698332E+00      -1.767155E-07      7.11321
1      SYMMETRIC THREE BAR TRUSS DESIGN OPTIMIZATION - DSOUG14      DECEMBER 18, 2001 MSC.NASTRAN 4/ 9/01
0      CONTINUATION - 3 CROSS SECTIONAL AREAS AS DESIGN VARIABLES
0

DESIGN VARIABLE HISTORY
-----
INTERNAL | EXTERNAL | LABEL | INITIAL : 1 : 2 : 3 : 4 :
DV. ID. | DV. ID. |       |       |
1 | 1 | A1 | 8.3652E-01 : 8.6093E-01 : 8.4220E-01 :
2 | 2 | A2 | 3.2941E-01 : 2.6353E-01 : 3.1623E-01 :
3 | 3 | A3 | 8.3652E-01 : 8.6093E-01 : 8.4220E-01 :
*** USER INFORMATION MESSAGE 6464 (DOM12E)
RUN TERMINATED DUE TO HARD CONVERGENCE TO AN OPTIMUM AT CYCLE NUMBER = 2.
```

Even though this example did not show a benefit from a subsequent submittal, it does illustrate several reasons why this type of resubmittal could be beneficial. These reasons and others are given in the following list:

1. Investigate the effect of optimization parameters.
2. Enable the ability to pause and examine the results, possibly adjusting the optimization parameters. This is particularly useful in large problems where it would be prudent to limit the number of design cycles until confidence is gained that the task is proceeding satisfactorily.
3. Investigate the effects of linking options.
4. Focus in on an intermediate design that is of interest. (This requires setting DESPCH=1 so that design results are captured after each design cycle).
5. Apply more analysis types and/or subcases to the final design from the initial design task.
6. Use the previously computed optimal design with modified design requirements.

An alternative use of the .pch file is to take the final property values from this file and create a data file that can be used for other types of analyses. For example, nonlinear analyses that cannot be performed in SOL 200 could now be run in order to check this behavior.

Continuing a Shape Optimization Design Task

The procedures are somewhat different for a shape optimization task since, in this case, the DESVAR values do not cause an override of the grid locations. Recall from [Equation \(6-5\)](#) on [page 312](#) that the



design change is the product of the shape basis vector and the *change in the design variable* values. If the technique of using the final DESVAR values in a resubmittal were applied, there would be no change in the shape relative to the initial design since the new design task knows nothing about the original DESVAR values. For shape optimization then, it is necessary to input the final grid locations for the new job submittal. Again, the .pch file can be used for this purpose since the new GRID bulk data entries are provided in the this file. The DESPCH parameter specifies the frequency of this output with the default punching the grid bulk data entries for the final design. (There is a special case when the optimization creates a mesh that provides distorted elements, making it impossible to proceed. In this case, the punch file contains both the last good mesh and the final distorted mesh.)

Restarting from a Previous Analysis

The second class of restarts is based on MSC 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 constraint evaluation and sensitivity analysis phases. This recalculation occurs regardless of whether the design model has changed or not. You are cautioned that it makes no sense to perform this type of restart when changing the design model because the analysis results from the previous run are no longer valid.

To illustrate this procedure, suppose we wanted to perform a baseline analysis for the structure of TPL file DSOUG1 (the first example in this chapter), compute and display the design responses 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 of [Listing 8-33](#) is the Bulk Data parameter OPTEXIT. In this case, OPTEXIT is set to 3 which instructs the code to exit after design constraint evaluation and screening.

Listing 8-33 Three Bar Truss with OPTEXIT=3 (DSOUG1C)

```
$
ID MSC  DSOUG1 $
TIME 10      $
SOL 200      $  OPTIMIZATION
CEND
TITLE = SYMMETRIC THREE BAR TRUSS DESIGN OPTIMIZATION -          DSOUG1
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
SPC1 100 123456 1      THRU      3
$ 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      NAME      PMIN      PMAX      C0      +
$+      DVVID1      COEF1      DVVID2      COEF2      ...
DVPREL1 10      PROD      11      A
1      1.0
DVPREL1 20      PROD      12      A
2      1.0
DVPREL1 30      PROD      13      A
3      1.0
$-
$...STRUCTURAL RESPONSE IDENTIFICATION
$DRESP1 ID      LABEL      RTYPE      PTYPE      REGION      ATTA      ATTB      ATT1      +
$+      ATT2      ...
DRESP1 20      W      WEIGHT
DRESP1 21      U4      DISP
DRESP1 23      S1      STRESS      PROD
12      13
$-
$...CONSTRAINTS
$DCONSTR DCID      RID      LALLOW      UALLOW
DCONSTR 21      21      -0.20      0.20

```



```

DCONSTR 21      23      -15000. 20000.
$ . . .
$ ...OPTIMIZATION CONTROL:
$ .
DOPTPRM DESMAX 10      DELP    0.5      P1      1      P2      15
PARAM   OPTEXIT 3
$ .
$ .....2.....3.....4.....5.....6.....7.....8.....9.....0
ENDDATA

```

After the finite element and design results are examined and verified, the decision is made to continue with optimization in the restart run. This file is shown in [Listing 8-34](#).

Listing 8-34 Restart File for Three Bar Truss (DSOUG1R)

```

RESTART, VERSION=1, KEEP
ID MSC DSOUG1 $
TIME 10      $
DIAG 8,15
SOL 200      $ OPTIMIZATION
CEND
TITLE = SYMMETRIC THREE BAR TRUSS DESIGN OPTIMIZATION -          DSOUG1
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
/
28
ENDDATA

```

The RESTART command is required, and indicates that the first version on the database is to be used (VERSION=1) and all versions are to be retained (KEEP). If we always intend to use the last version on the database for restarts and delete the rest, we can instead use,

```
RESTART, VERSION=LAST, NOKEEP
```

By specifying VERSION=1, KEEP, we always have the first version to fall back on.

The original database can be selected in one of two ways. At job submittal, the database can be selected with a command such as:

```
nastran dsoug1r scr=n dbs=dsoug1c
```

alternatively, an assign statement can be included in the data file:

```
ASSIGN MASTER=dsoug1c.MASTER
```

In [Listing 8-34](#), the `/,28' deletes the OPTEXIT parameter assignment from the Bulk Data, which is entry number 28 in the sorted bulk data of the original run. It is usually a good idea to request a listing of the



sorted Bulk Data, especially if subsequent restarts are anticipated. Sorted bulk data is the default, but is explicit with the Case Control command, ECHO=SORT or ECHO=BOTH.

The optimization results from this restart are identical to those given in [Listing 8-2 on page 495](#) and are not repeated here. The contents of the .F04 file include statements such as the following, which indicate the initial analysis was skipped.

10:50:54	0:03	237.0	0.0	3.1	0.0	PHASE1A 70	(S)SEMG	BEGN
56							SEMG	NOT EXECUTED.
10:50:54	0:03	239.0	0.0	3.1	0.0	PHASE1A 134	(S)SELG	BEGN
56							SELG	NOT EXECUTED.



Cantilevered Plate with Redundant Materials

This section uses a contrived variation of the DSOUG3 (Cantilevered Plate) example to demonstrate the use of the DRESP1 that selects weight as a function of material ID.

The finite element representation is essentially the same as that of the DSOUG3 example, but in this DSOUG14 example, two sets of elements are positioned connecting the same grids. One set of elements is aluminum while the second is steel and the design task is to use DRESP1 RTYPE=WMPID responses in a DRESP2 to construct a cost function that incorporates the weight and the unit cost of each material.

$$\text{Total Cost} = \text{Unit Cost of Aluminum} * \text{Weight of Aluminum} + \text{Unit Cost of Steel} * \text{Weight of Steel}$$

Other differences from the DSOUG3 case are that only the static analyses are applied and the design variables directly design the element thicknesses rather than via the reduced basis feature of DSOUG3.

Analysis Model Description

2 x 8 array of CQUAD4 elements

Property/Material	Aluminum	Steel
E	1.0E7	3.0E7
Poisson's Ratio	.33	.33
Weight Density	.1 lb./in ³	.3 lb./in ³

Two static load conditions:

Tip loading:

Two 500 lb loads in -z direction

Pressure loading:

Uniform at 7.5 lb/in²



Design Model Description

Objective: Structural weight minimization

Design variables: Eight design variables control the thicknesses of the spanwise aluminum elements and eight other design variables control the thicknesses of the spanwise steel elements.

Constraints:

ANALYSIS = STATICS	$-2.0 \leq$ Tip Displacement ≤ 2.0
	$Von Mises Stress \leq 29\text{ksi}$

Listing 8-35 shows the input for DSOUG14. The file is similar to that of Listing 8-5, so this discussion will focus on the unique features of this deck. As mentioned, two sets of elements are superimposed, one with an aluminum material and one with steel. Sixteen design variables control the thickness of the 32 elements (elements of the same material and span station have the same design). Constraints are placed on the displacements and stresses under the two loading conditions. The objective is a DRESP2 that is a combination of two DRESP1's that give the weight of the separate materials and DTABLE entries that provide a relative cost of the material. In the example of the listing, the cost of the aluminum (CAL) is set to 6.0 while the cost of steel (CST) is set at 1.0.

Listing 8-35 Input File for DSOUG14

```

ID MSC  DSOUG14 $
TIME 10
SOL 200      $  OPTIMIZATION
CEND
TITLE      = CANTILEVERED PLATE - DSOUG14
SUBTITLE   = COST AS A WEIGHTED SUM OF MATERIAL WEIGHTS
SPC       = 100
DISP      = ALL
STRESS     = ALL
DESOBJ(MIN) = 350  $ OBJECTIVE FUNCTION DEFINITION
dsaprt(start=1,by=1)
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
MAT1      52        3.0E+7          0.33      0.3          +M2
+M2      50000.    50000.    29000.    11           21
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	11
CQUAD4	2	2	2	3	12
CQUAD4	3	3	3	4	13
CQUAD4	4	4	4	5	14
CQUAD4	5	5	5	6	15
CQUAD4	6	6	6	7	16
CQUAD4	7	7	7	8	17
CQUAD4	8	8	8	9	18
CQUAD4	11	1	11	12	21
CQUAD4	12	2	12	13	22
CQUAD4	13	3	13	14	23
CQUAD4	14	4	14	15	24
CQUAD4	15	5	15	16	25
CQUAD4	16	6	16	17	26
CQUAD4	17	7	17	18	27
CQUAD4	18	8	18	19	28
CQUAD4	101	11	1	2	11
CQUAD4	102	21	2	3	12
CQUAD4	103	31	3	4	13
CQUAD4	104	41	4	5	14
CQUAD4	105	51	5	6	15
CQUAD4	106	61	6	7	16
CQUAD4	107	71	7	8	17
CQUAD4	108	81	8	9	18
CQUAD4	111	11	11	12	21
CQUAD4	112	21	12	13	22
CQUAD4	113	31	13	14	23
CQUAD4	114	41	14	15	24
CQUAD4	115	51	15	16	25
CQUAD4	116	61	16	17	26
CQUAD4	117	71	17	18	27
CQUAD4	118	81	18	19	28
PSHELL	1	51	3.0	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	



```

PSHELL   11      52      3.0      51
PSHELL   21      52      2.640625 51
PSHELL   31      52      2.3125   51
PSHELL   41      52      2.015625 51
PSHELL   51      52      1.75     51
PSHELL   61      52      1.515625 51
PSHELL   71      52      1.3125   51
PSHELL   81      52      1.140625 51
$        2       3j      4       5       6       7       8       9       10
FORCE    300     9       500.    0.0     0.0     -1.0
FORCE    300     29      500.    0.0     0.0     -1.0
PLOAD2   310    7.5      1       THRU    8
PLOAD2   310    7.5     11      THRU    18
$
$-----
$ DESIGN MODEL:
$-----
$...DEFINE THE DESIGN VARIABLES
$DESVAR ID      LABEL      XINIT      XLB      XUB      DELXV
DESVAR10t1a  1.0.001
DESVAR20t2a  1.0.001
DESVAR30t3a  1.0.001
DESVAR40t4a  1.0.001
DESVAR50t5a  1.0.001
DESVAR60t6a  1.0.001
DESVAR70t7a  1.0.001
DESVAR80t8a  1.0.001
DESVAR110t1s 1.0.001
DESVAR120t2s 1.0.001
DESVAR130t3s 1.0.001
DESVAR140t4s 1.0.001
DESVAR150t5s 1.0.001
DESVAR160t6s 1.0.001
DESVAR170t7s 1.0.001
DESVAR180t8s 1.0 .001
$
$...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       T
101.0
DVPREL1 2      PSHELL   2       T
201.0
DVPREL1 3      PSHELL   3       T
301.0
DVPREL1 4      PSHELL   4       T
401.0
DVPREL1 5      PSHELL   5       T
501.0
DVPREL1 6      PSHELL   6       T
601.0
DVPREL1 7      PSHELL   7       T
701.0
DVPREL1 8      PSHELL   8       T
801.0
DVPREL1 11     PSHELL  11      T
1101.0
DVPREL1 12     PSHELL  21      T
1201.0
DVPREL1 13     PSHELL  31      T
1301.0
DVPREL1 14     PSHELL  41      T
1401.0
DVPREL1 15     PSHELL  51      T
1501.0
DVPREL1 16     PSHELL  61      T

```



```

1601.0
DVPREL1 17      PSHELL  71      T
1701.0
DVPREL1 18      PSHELL  81      T
1801.0
$
$...IDENTIFY THE DESIGN RESPONSES
$DRESP1 ID      LABEL    RTYPE   PTYPE   REGION  ATTA   ATTB   ATT1
$+      ATT2    ...
$      STATIC VON MISES STRESS IN SHELL ELEMENTS
DRESP1  2       S12     STRESS ELEM          9           1
2345678
DRESP1  20      S12     STRESS ELEM          9        101
102103104105106107108
$      STATIC DISPLACEMENT AT THE TIP
DRESP1  33      D1      DISP             3        19
$
$...DEFINE THE WEIGHT RESPONSE TO BE USED AS THE OBJECTIVE FUNCTION:
DRESP1 35      W      WEIGHT
DRESP1 150     WAL    wmpid51
DRESP1 250     Wst    wmpid52
dresp2350cost350
dtablecalcst
dresp1150250
deqatn350cost(cal,cst,wal,wst) = cal * wal + cst*wst
dtablecal6.0cst1.0
$
$...DEFINE THE STATIC DESIGN CONSTRAINTS
$DCONSTR DCID   RID    LALLOW UALLOW
DCONSTR 10      2      -29000. 29000.
DCONSTR 10      20     -1.0e5  1.0e5
DCONSTR 10      33     -2.      2.
$
$...OVERRIDE OPTIMIZATION PARAMETER DEFAULTS (OPTIONAL)
DOPTPRM DESMAX 20      P11P215  iprint7
ct-.10
$
PARAMNAPSRT1
PARAMPOST-1
ENDDATA

```

Results in terms of the design history and are shown in Listing 8-36. It is seen that for this example the design takes the aluminum thicknesses to their lower gage and the steel material has a tapered design to satisfy the imposed constraints. This design arises from the fact that the aluminum material is high cost and therefore avoided. A different answer would be obtained if the two costs were the same.

Listing 8-36 Design History DSOU14

***** 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			21	
NUMBER OF OPTIMIZATIONS W.R.T. APPROXIMATE MODELS			20	
***** O B J E C T I V E A N D M A X I M U M C O N S T R A I N T H I S T O R Y *****				
CYCLE NUMBER	OBJECTIVE FROM APPROXIMATE OPTIMIZATION	OBJECTIVE FROM EXACT ANALYSIS	FRACTIONAL ERROR OF APPROXIMATION	MAXIMUM VALUE OF CONSTRAINT
INITIAL	3.60000E+02	2.45804E+02	-9.938520E-05	-3.039457E-01
1	2.45780E+02	2.45804E+02	1.477469E-05	1.720341E-01
2	2.168840E+02	2.168808E+02	1.477112E-01	-1.687112E-01
3	1.709618E+02	1.709470E+02	8.613622E-05	-5.324024E-02
4	1.390167E+02	1.390016E+02	1.089360E-04	5.885601E-02
5	1.298517E+02	1.298456E+02	4.688844E-05	7.933152E-02
6	1.276995E+02	1.276974E+02	1.619113E-05	3.830230E-02
7	1.251049E+02	1.251046E+02	1.951492E-06	2.410996E-02
8	1.223171E+02	1.223165E+02	4.490942E-06	1.692879E-02
9	1.193379E+02	1.193380E+02	-1.150757E-06	1.640475E-02
10	1.163069E+02	1.163067E+02	1.639930E-06	1.381922E-02
11	1.134235E+02	1.134226E+02	7.399171E-06	9.416699E-03



12		1.127242E+02	1.127253E+02	-1.008451E-05	1.168334E-02
13		1.097636E+02	1.097708E+02	-1.049495E-05	2.390134E-02
14		1.099024E+02	1.099017E+02	5.761872E-06	1.067364E-02
15		1.101128E+02	1.101142E+02	-1.281795E-05	7.457495E-03
16		1.099600E+02	1.099609E+02	-7.840259E-06	1.006174E-02
17		1.095199E+02	1.095202E+02	-2.368508E-06	8.524537E-03
18		1.095344E+02	1.095347E+02	-2.325416E-06	6.943583E-03
19		1.095872E+02	1.095871E+02	1.044292E-06	4.624126E-03
20		1.096039E+02	1.096040E+02	-1.044131E-06	3.609657E-03

DESIGN VARIABLE HISTORY

INTERNAL DV. ID.	EXTERNAL DV. ID.	LABEL	INITIAL	1	2	3	4	5
1	10	T1A	1.00000E+00	9.1998E-01	4.6414E-01	2.3207E-01	1.1604E-01	8.7062E-02
2	20	T2A	1.00000E+00	8.2197E-01	4.6642E-01	2.3321E-01	1.1660E-01	8.7475E-02
3	30	T3A	1.00000E+00	7.0309E-01	4.5512E-01	2.2756E-01	1.1378E-01	8.5347E-02
4	40	T4A	1.00000E+00	5.5396E-01	4.3490E-01	2.1745E-01	1.0873E-01	8.1553E-02
5	50	T5A	1.00000E+00	5.0000E-01	5.0306E-01	2.5153E-01	1.2576E-01	9.4348E-02
6	60	T6A	1.00000E+00	5.0000E-01	5.4616E-01	2.7985E-01	1.3993E-01	1.0499E-01
7	70	T7A	1.00000E+00	5.0054E-01	3.3835E-01	2.4922E-01	1.2461E-01	9.3536E-02
8	80	T8A	1.00000E+00	5.0006E-01	2.5003E-01	2.1144E-01	1.4404E-01	1.0803E-01
9	110	T1S	1.00000E+00	1.2919E+00	1.3017E+00	1.7186E+00	1.0953E+00	1.3691E+00
10	120	T2S	1.00000E+00	1.1555E+00	1.2990E+00	1.4115E+00	1.1496E+00	1.3483E+00
11	130	T3S	1.00000E+00	9.9498E-01	1.2930E+00	1.1142E+00	1.2031E+00	1.0845E+00
12	140	T4S	1.00000E+00	8.1034E-01	1.2155E+00	9.3440E-01	1.1516E+00	9.3064E-01
13	150	T5S	1.00000E+00	6.3502E-01	9.5253E-01	8.8615E-01	9.6405E-01	8.3842E-01
14	160	T6S	1.00000E+00	5.0000E-01	7.5000E-01	6.8167E-01	8.3935E-01	6.9979E-01
15	170	T7S	1.00000E+00	5.0000E-01	4.8058E-01	5.9212E-01	5.9683E-01	5.7264E-01
16	180	T8S	1.00000E+00	5.0000E-01	2.5000E-01	2.5306E-01	2.8790E-01	3.2823E-01
INTERNAL DV. ID.	EXTERNAL DV. ID.	LABEL	6	7	8	9	10	11
1	10	T1A	7.4562E-02	6.2064E-02	4.9564E-02	3.7064E-02	2.4564E-02	1.2064E-02
2	20	T2A	7.4975E-02	6.2477E-02	4.9977E-02	3.7477E-02	2.4977E-02	1.2477E-02
3	30	T3A	7.2847E-02	6.0349E-02	4.7849E-02	3.5349E-02	2.2849E-02	1.0349E-02
4	40	T4A	6.9053E-02	5.6554E-02	4.4054E-02	3.1554E-02	1.9054E-02	6.5533E-03
5	50	T5A	6.1848E-02	6.9350E-02	5.6850E-02	4.4350E-02	3.1850E-02	1.9305E-02
6	60	T6A	6.1666E-02	7.9322E-02	6.6866E-02	5.4372E-02	4.1872E-02	2.3372E-02
7	70	T7A	8.1020E-02	6.9354E-02	5.4340E-02	4.1540E-02	3.1440E-02	1.9450E-02
8	80	T8A	9.4527E-02	8.2098E-02	6.9598E-02	5.7098E-02	4.4598E-02	3.2098E-02
9	110	T1S	1.2110E+00	1.3622E+00	1.2352E+00	1.3650E+00	1.2587E+00	1.3228E+00
10	120	T2S	1.1798E+00	1.2874E+00	1.1846E+00	1.2661E+00	1.2172E+00	1.2479E+00
11	130	T3S	1.1288E+00	1.1258E+00	1.1332E+00	1.0994E+00	1.1529E+00	1.1344E+00
12	140	T4S	1.0470E+00	9.8865E-01	1.0639E+00	9.9045E-01	1.0483E+00	9.9907E-01
13	150	T5S	9.4322E-01	8.6501E-01	9.4510E-01	8.6677E-01	9.3020E-01	8.7321E-01
14	160	T6S	7.8726E-01	7.2793E-01	7.9419E-01	7.3952E-01	7.7707E-01	7.4691E-01
15	170	T7S	6.3542E-01	5.6491E-01	6.0312E-01	5.9652E-01	5.7889E-01	6.0818E-01
16	180	T8S	2.9938E-01	3.3681E-01	3.1340E-01	3.5053E-01	3.0887E-01	3.4748E-01
INTERNAL DV. ID.	EXTERNAL DV. ID.	LABEL	12	13	14	15	16	17
1	10	T1A	1.0620E-02	1.0000E-03	1.0000E-03	1.0000E-03	1.0000E-03	1.0000E-03
2	20	T2A	1.0932E-02	1.0000E-03	1.0000E-03	1.0000E-03	1.0000E-03	1.0000E-03
3	30	T3A	9.2865E-03	1.0000E-03	1.0000E-03	1.0000E-03	1.0000E-03	1.0000E-03
4	40	T4A	6.1276E-03	1.0000E-03	1.0000E-03	1.0000E-03	1.0000E-03	1.0000E-03
5	50	T5A	1.5638E-02	3.1380E-03	3.1141E-03	3.0781E-03	3.0557E-03	3.0000E-03
6	60	T6A	2.0824E-02	8.3236E-02	8.1550E-02	7.9080E-02	7.7604E-02	1.5104E-03
7	70	T7A	1.5135E-02	2.6352E-03	2.6184E-03	2.5929E-03	2.5770E-03	1.0000E-03
8	80	T8A	2.1920E-02	9.4280E-02	9.1620E-02	8.8910E-02	8.7640E-02	2.4460E-02
9	110	T1S	1.2177E+00	1.3692E+00	1.2842E+00	1.1469E+00	1.2641E+00	1.3346E+00
10	120	T2S	1.1847E+00	1.1597E+00	1.1982E+00	1.2668E+00	1.1877E+00	1.2619E+00
11	130	T3S	1.1349E+00	1.1380E+00	1.1308E+00	1.1348E+00	1.1294E+00	1.1322E+00
12	140	T4S	1.0646E+00	9.8527E-01	1.0469E+00	9.9409E-01	1.0506E+00	1.0084E+00
13	150	T5S	9.5846E-01	8.5191E-01	9.0516E-01	8.8907E-01	9.2109E-01	8.8956E-01
14	160	T6S	8.0696E-01	7.2463E-01	7.6992E-01	7.4380E-01	7.8441E-01	7.5082E-01
15	170	T7S	6.0232E-01	5.8131E-01	6.0724E-01	5.7388E-01	6.0732E-01	5.8172E-01
16	180	T8S	3.2437E-01	3.5226E-01	3.3024E-01	3.3780E-01	3.3394E-01	3.2231E-01
INTERNAL DV. ID.	EXTERNAL DV. ID.	LABEL	18	19	20	21	22	23
1	10	T1A	1.0000E-03	1.0000E-03	1.0000E-03	1.0000E-03	1.0000E-03	1.0000E-03
2	20	T2A	1.0000E-03	1.0000E-03	1.0000E-03	1.0000E-03	1.0000E-03	1.0000E-03
3	30	T3A	1.0000E-03	1.0000E-03	1.0000E-03	1.0000E-03	1.0000E-03	1.0000E-03
4	40	T4A	1.0000E-03	1.0000E-03	1.0000E-03	1.0000E-03	1.0000E-03	1.0000E-03
5	50	T5A	1.0000E-03	1.0000E-03	1.0000E-03	1.0000E-03	1.0000E-03	1.0000E-03
6	60	T6A	1.4999E-03	1.4817E-03	1.4752E-03	1.0000E-03	1.0000E-03	1.0000E-03
7	70	T7A	1.0000E-03	1.0000E-03	1.0000E-03	1.0000E-03	1.0000E-03	1.0000E-03
8	80	T8A	2.4331E-03	2.3854E-03	2.3684E-03	1.0000E-03	1.0000E-03	1.0000E-03
9	110	T1S	1.2770E+00	1.3048E+00	1.2918E+00	1.2276E+00	1.2276E+00	1.2276E+00
10	120	T2S	1.2012E+00	1.2291E+00	1.2276E+00	1.2276E+00	1.2276E+00	1.2276E+00
11	130	T3S	1.1233E+00	1.1402E+00	1.1313E+00	1.1313E+00	1.1313E+00	1.1313E+00
12	140	T4S	1.0342E+00	1.0508E+00	1.0217E+00	1.0217E+00	1.0217E+00	1.0217E+00
13	150	T5S	9.1601E-01	9.1177E-01	9.0445E-01	9.0445E-01	9.0445E-01	9.0445E-01
14	160	T6S	7.8080E-01	7.6533E-01	7.6667E-01	7.6667E-01	7.6667E-01	7.6667E-01
15	170	T7S	6.0885E-01	5.8107E-01	6.0615E-01	6.0615E-01	6.0615E-01	6.0615E-01
16	180	T8S	3.2446E-01	3.2105E-01	3.3991E-01	3.3991E-01	3.3991E-01	3.3991E-01

*** USER INFORMATION MESSAGE 6464 (DOM12E)
 RUN TERMINATED DUE TO HARD CONVERGENCE TO AN OPTIMUM AT CYCLE NUMBER = 20.





9

Special Topics

- Introduction 606
- Discrete Variable Optimization 606
- Fully Stressed Design 608
- Trust Region 611
- Superelement Optimization 617
- Automatic External Superelement Optimization (AESO) 622
- Randomization of a User's Input Data File 631
- Optimization of Nonlinear Structural Responses (Pre-release) 633
- Solving Large Problems in SOL 200 655
- Special Considerations When Designing One-Dimensional Bending Elements 660
- MultiOpt-Global Optimization and Multi Model Optimization 664
- OpenMDOTM 679



Introduction

This Chapter deals with special topics that are definitely part of MSC Nastran's design optimization capability, but do not fit neatly into one of the previous chapters.

Discrete Variable Optimization

Most optimization algorithms are based on the assumption that the design variables or quantities that are modified in search of an optimal design can vary continuously within a defined range. Such is the case with the gradient-based algorithms that are the focus of this User's Guide. However, the practicing engineer must often choose from discrete values that most closely match the optimal vector of design variables returned by the optimizer. The constraints of mass production usually do not afford the engineer the luxury of specifying a 1.84 mm plate thickness; instead the engineer must make a practical choice from among several of the nearest available gauge sizes. A number of subsequent analyses will often need to be performed to verify that the design goals have been reasonably met and that none of the critical performance constraints have been violated. Discrete variable optimization seeks to avoid such tedium by allowing the engineer to specify a list of available gauges beforehand, leaving the task of selection and validation to the optimizer.

Although some theoretical algorithms promise a high degree of precision (such as the Branch and Bound Method), most algorithms are not computationally efficient and therefore, are unsuitable for use with large-scale finite element analysis programs such as MSC Nastran. Instead, four practical and cpu-efficient methods have been implemented:

1. Rounding up to the nearest discrete design variable
2. Rounding off to the nearest discrete design variable
3. Conservative Discrete Design (CDD)
4. Design of Experiments (DOE)

The first two methods simply automate the simple rounding process a user might employ after a continuous optimization and require no new analyses. For the CDD method, each variable is independently set to the discrete values that bracket the continuous variable result. An approximate analysis is carried out for the discrete variable above the continuous value and one with the discrete variable below the continuous value. The constraint results of these two analyses are compared and the discrete variable is chosen that gives the minimum value for the maximum constraint. This is repeated for each design variable so that 2^{nddv} (where nddv is the number of design variables that can take on discrete sizes) approximate analyses are carried out for the CDD approach.

The DOE method implemented in MSC Nastran makes use of the concept of orthogonal arrays. As with the CDD method, the postprocessing task is limited to searching for designs where only the discrete variable values just above and just below the continuous variable are selected. In this case, however, any combination of discrete variables can be used. This seemingly straightforward task can still be difficult as there are 2^{nddv} possible vectors that could be analyzed. With a modest 25 variables, $3.35 \times 10^7 (2^{25})$ functions evaluations would be required for an exhaustive search. At 10 evaluations per second, this would require over a month of computing and is clearly impractical. The implementation of DOE employed in MSC Nastran employs an exhaustive search when nddv is 16 or less. Above this value, an



Orthogonal Array concept is employed to select 2^{16} candidate arrays that provide a representative sampling of the overall design space. Clearly this is an approximation, but the thinking is that this will provide adequate coverage of the possible discrete solutions that it will not be far off from the “true” optimum that would be obtained from an exhaustive search.

Each of these methods is a postprocessing operation of the continuous, optimal design. Moreover, the engineer can specify that such discrete variable postprocessing be applied only to the final design, or to any number of the previous, intermediate designs that lead up to the final design. Useful design information can often be obtained by observing the progression of (hopefully) converging “optimal” discrete values, because any one of the intermediate solutions may actually result in a more usable design than the optimal solution itself. Note that computation of these intermediate discrete solutions has no influence on subsequent designs because the next design cycle is always based on the previous continuous solution and not the discrete solution.

Discrete variables can also be mixed with continuous ones, allowing “mixed-discrete” optimization. A typical example is that of combined sizing and shape optimization; gauge sizes must be discrete, while the variables that control shape may vary smoothly and continuously. Furthermore, both mathematical programming (MP) and fully stressed design approaches can be used in conjunction with discrete variable optimization.



Fully Stressed Design

Fully Stressed Design (FSD) is a concept that has long been regarded as an automated design algorithm that can quickly produce a design that satisfies key design requirements. Its implementation in MSC Nastran complements the long standing mathematical-programming (MP) approach to automated design that is the focus of this User's Guide. It utilizes many of the same procedures to produce a "quick-look" design at a small fraction of the computational costs. This efficiency can be exploited by requesting that the redesign take place with literally thousands of independently designed properties, whereas an MP approach is realistically limited to approximately one thousand design variables. The method is particularly useful in the design of aerospace structures where the overriding requirement is that the structural weight be minimized. Although the proposed design that results from the FSD technique may not be usable from a manufacturing standpoint because it has neglected key design considerations, it can be thought of as providing a lower bound estimate on the amount of structural material required to achieve the imposed design conditions. The output of the FSD algorithm can also be used as an excellent starting design for a more general MP design task.



The basic resizing concept of FSD can be described as follows

$$t_i^{\text{new}} = (\sigma_i / \sigma_{\text{all}})^{\alpha} t_i^{\text{old}} \quad (9-1)$$

where

- t = designed property
- i = index to indicate which property contains the design parameter and the design response
- $\sigma, \sigma_{\text{all}}$ = actual and allowable response quantities, such as stress
- α = a real number ($0.0 < \alpha \leq 1.0$)

And the *old* and *new* superscripts refer to before and after the resizing.

The implementation of FSD within MSC Nastran is embedded within the SOL 200 Design Optimization. Design variable and constraint conditions are applied using the existing Bulk Data entries and Case Control commands that have been developed for the MP algorithm. Two parameters control the FSD algorithm:

FSDALP	Relaxation parameter applied in Fully Stressed Design. (Real, $0.0 < \text{FSDMAX} \leq 1.0$, Default = 0.9)
FSDMAX	Specifies the number of Fully Stressed Design Cycles that are to be performed. (Integer, Default = 0)

The FSD algorithm shares much of its user interface and dataflow with the basic design optimization capability. The capabilities available for FSD are a subset of those available for general multidisciplinary optimization and include:

1. FSD is applicable to static and static aeroelastic analyses. Other analyses can be included in the input and will be evaluated in the analysis associated with the FSD cycles and included in the redesign in any subsequent MP design cycles.
2. Multiple subcases and multiple boundary conditions are supported.
3. Composite materials are supported.
4. Allowable limits can be placed on element stress and/or strain.
5. PROD areas and shell thicknesses s (PSHELL, PSHEAR and PCOMP thicknesses) are supported with FSD.
6. For composites, the thicknesses of individual PCOMP layers can be resized using the FSD algorithm.
7. All elements sharing a single property ID are sized to a single value that satisfies all the imposed conditions. Further, if a number of properties are controlled by a single DESVAR, the single DESVAR value will be sized to satisfy all imposed conditions.



8. Minimum and maximum size limits are imposed (e.g., design variable bounds and property limits).

The above list implies the following additional limitations:

1. Beam cross-sections are not designed. Ply orientation angle cannot be used as a design variable for FSD.
2. If an element is constrained, but the corresponding property for the element is not resized, the constraint is ignored in the resizing.
3. If a property is designed, but has no imposed constraints, the property value is unchanged by the resizing algorithm.
4. Shape design variables and material and connectivity properties are not supported in FSD.
5. Each designed property can reference only a single design variable.

A guideline for the use of the FSD capability is to limit its application to 5-10 design cycles and follow it up, if practical, with MP design cycles. This is performed seamlessly in SOL 200 by the resizing algorithm switching from FSD to MP after FSDMAX design cycles or after the FSD design is considered converged. The MP algorithms have much more generality in application and are also known to be able to achieve a lower weight structure. This is not practical when the FSD design task has thousands of design variables since this is likely to overwhelm the MP algorithm. The default value of 0.9 for FSDALP is usually adequate to achieve quick convergence. Lower values, such as 0.5, can be used to achieve a completely converged solution.



Trust Region

Introduction

Formal approximate optimization is used in SOL 200 to find a new design without performing expensive and exact finite element analyses ([The Approximate Model](#)). Move limits must be applied to define a restricted region on which an adequate approximate model can be created.

Convergence checks are made following each approximate optimization task. Currently, if the approximate optimization comes up with a design that an exact analysis shows is actually worse, it is assumed that the region defined by move limits are too generous and they are reduced. Using Trust Region Concepts, this simple update strategy can be improved in three fundamental ways:

1. A merit function is applied to provide a more quantitative measure of the quality of the approximation.
2. A provision is made to increase the move limits when the approximate optimization results are shown to be of high quality.
3. If the new design is judged to actually be worse, it is discarded and the approximate optimization task is performed again with tighter move limits.

Benefits

Since the move limits can be either reduced or increased in an adaptive fashion, it is expected that the quality of the approximate model will be enhanced. This should lead to more robust optimization result and faster convergence. In addition, rejecting a bad design should also smooth the design optimization process.

Theory

Concept of Trust Region

The main idea behind the Trust Region is to first compute a merit function that is a combination of the objective function and the maximum violated constraint value and then form a ratio of the exact reduction in a merit function to the predicted reduction in the merit function. The ratio can be written as follows, assuming a design task is a constrained minimization optimization task:

$$\text{Ratio} = \frac{MF_p - MF_c}{MF_p - MF_a}$$

where:

$$MF_c = \Phi_c + PW \cdot \max g_c$$

The current exact merit function

$$MF_p = \Phi_p + PW \cdot \max g_p$$

The previous exact merit function

$$MF_a = \Phi_a + PW \cdot \max g_c$$

The approximate merit function.



Notice that a merit function is the sum of objective (Φ) and the product of penalty weight and the maximum violated constraint (g). Subscripts c, p and a, indicate that the function is evaluated exactly at the current design cycle, evaluated exactly at the previous design cycle and evaluated approximately, respectively. The penalty weight plays an important role in a successful optimization task with Trust Region Method.

The magnitude of Ratio varies in the real interval of $(-\infty, \infty)$. The denominator is the predicted reduction in merit function while the numerator is the exact reduction in merit function. Since we consider the case of minimizing the objective, the denominator should always produce a positive number to indicate that the optimizer has done a good job.

Different Ratio values can measure the quality of the approximate model. For example, a negative Ratio (likely due to negative numerator) indicates that the approximate model is so bad that the exact merit function is greater than the previous exact. Thus reducing move limits is necessary. On the other hand, Ratio = 1 indicates the approximate model is very good that the predicted reduction is identical to the exact reduction and the move limits can be increased. In general, the following strategies are used to update move limits:

If $\text{Ratio} \leq \eta_1$ (say 0.01), reject the design, cut the move limits in half and repeat the approximate optimization task. For a minimization task, the reduction in the merit function must be greater than zero. A too small or negative Ratio indicates that the new design proposed by the approximate model either does not reduce the merit function, or the quality of the approximate model is very poor.

If $\eta_1 < \text{Ratio} \leq \eta_2$ (say 0.25), accept the design, but cut the move limits in half for the next approximate design task. The ratio in this range indicates that quality of the approximate model is not good enough, or the current move limit is too large.

If $\eta_2 < \text{Ratio} \leq \eta_3$ (say 0.75), accept the design and do not adjust the move limits.

If $\text{Ratio} > \eta_3$, accept the design, but increase the move limits by 50%, up to a pre-specified upper value. The ratio approaching to 1 indicates that the approximate model is accurately predicting the actual behavior. Therefore, the current move limits can be increased. In addition, although $\text{Ratio} > 1$ indicates that the approximate model is not particularly accurate, but it predicts in the right direction so that larger move limits can also be used.

Penalty Weight

Penalty Weight (PW) is used to form the merit function. The penalty weight plays an important role in a successful optimization task with Trust Region Method. It is problem dependent. The larger it becomes, the more the maximum violated constraint is penalized. There are two ways to define PW: automatic and user specified. For the automatic way, $PW = 10 * \text{the initial objective from the initial design cycle}$. This is the default option. Alternatively, the user can provide his own through a Bulk Data Parameter, MXLAGM1. Then, $PW = 10.* MXLAGM1$.



Rejecting a Design

If $\text{Ratio} \leq \eta_1$, the current design will be rejected. When the flag is set, the subsequent design sensitivity phase will be skipped and the approximate optimization job will be performed based on the previous good design with reduced move limits.

Input

Parameters related to Trust Region in DOPTPRM entry

TREGION	Flag to invoke Trust Region method. = 0 (Default) Don't employ trust regions = 1 turn Trust Region on
ETA1 (η_1)	the cutting ratio 1 (Default = 0.01), used by Trust Region Method.
ETA2 (η_2)	the cutting ratio 2 (Default = 0.25), used by Trust Region Method.
ETA3 (η_3)	the cutting ratio 3 (Default = 0.7), used by Trust Region Method.
UPDFAC1	Updating Factor 1 (Default = 2.0), used by Trust Region Method.
UPDFAC2	Updating factor 2 (Default = 0.5), used by Trust Region Method.
DPMAX	Maximum fraction of change on designed property (Default = 0.5), used by Trust Region Method.
DXMAX	Maximum fraction of change on design variable (Default = 1.0), used by Trust Region Method.

Bulk Data Parameter

PARAM,MXLAGM1,VALUE - Control parameter for Penalty Weight (Real, Default = 0.0).

Outputs

Special Printout from DOM12

If Trust Region method is invoked, DOM12 prints out additional information message as shown below. This message is printed out once every design cycle. The left column is design cycle number. It prints out Objective, Maximum violated constraint and merit function for exact current, exact previous and approximate. It further prints out Ratio together with numerator, denominator. It also prints out the final Update factor for move limits plus the rejecting status. Finally, it prints out current move limits for property and design variable plus penalty weight.



DCYCL	OBJECTIVE	MAX-CONS	MERIT-FUN	NUMER	DENOM	RATIO	UPDFAC	REJ-FLAG	TTTT
2	PREV 6.893711E+00	2.133793E+01	1.477869E+03	5.0832E+02	5.0839E+02	9.9987E-01	2.0000E+00	0	TTTT
CURR	9.935308E+00	1.392006E+01	9.695438E+02						TTTT
APPX	9.935904E+00	1.391907E+01	9.694761E+02						TTTT
MOVEP,MOVCP-MIN =	5.000000E-01	1.000000E-02							TTTT
MOVEX,MOVX-MIN =	1.000000E+00	5.000000E-02							TTTT
NDOFs (NDV-NACS) =	0								TTTT
PENAL WEIGHT =	0.6894E+02								TTTT

Marker Indicating a Rejected Design in Design History

A rejected design will be marked by adding symbol ‘R’ to the design cycle number shown in the Summary of Design Cycle History at the end of f06 file. Below is an example that shows design cycle 2 has been marked with ‘R’. This is because although this design produces a smaller objective but with the violated constraint. So the design is rejected. Design cycle 1 is used as the next starting design point.

***** 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				16
NUMBER OF OPTIMIZATIONS W.R.T. APPROXIMATE MODELS				15
OBJECTIVE AND MAXIMUM CONSTRAINT HISTORY				
-----	-----	-----	-----	-----
CYCLE NUMBER	OBJECTIVE FROM APPROXIMATE OPTIMIZATION	OBJECTIVE FROM EXACT ANALYSIS	FRACTIONAL ERROR OF APPROXIMATION	MAXIMUM VALUE OF CONSTRAINT
-----	-----	-----	-----	-----
INITIAL		6.614414E+02		1.102773E-01
1	5.987441E+02	5.987546E+02	-1.743120E-05	-4.438843E-04
2R	5.475672E+02	5.475823E+02	-2.764282E-05	2.106060E+00
3	5.626934E+02	5.627052E+02	-2.104267E-05	-3.308058E-04

Guidelines and Limitations

1. All these parameters in the DOPTPRM have their own defaults and are usually adequate. If you want the optimization job to have more aggressive move limit, you may do so by reduce ETA3 and/or increase DPMAX and DXMAX or increase UPDFAC1. Conversely, you may increase ETA3 and/or reduce DPMAX, DXMAX and/or decrease UPDFAC1.
2. A non-zero MXLAGM1 is used to override the default Penalty Weight (PW). This parameter is problem dependent and may require some iteration to come up a good number. When the need arises to specify one, a general rule is to specify a big number (say 100. and larger) when the maximum constraint is grossly violated.
3. In some cases, it is possible that activating Trust Region in SOL 200 may produce less desirable result than a non-Trust Region job.



Example

The example (dstrustr2.dat) is a Stiffened Panel constrained optimization job taken from the testing problem library (d200x7.dat). It minimizes a weight response while satisfying stress and displacement constraints. The Trust Region is invoked by specifying TREGION=1 on the DOPTPRM entry. In addition, the parameter MXLAGM1 is set to 10.

[Figure 9-1](#) plots the objective history and DELX history with and without TR. The TR job converges at design cycle 15 (notice that plots count initial design cycle as 1 that is actually zero if counted in the f06) while the non-TR job converges at design cycle 27. Both have same final objective function with maximum feasible constraint at the tolerance of GMAX=0.5%. The fast convergence achieved by the TR job may be explained by the DELX history shown at the bottom of [Figure 9-1](#). The non-trust region job uses a constant move limit (e.g., DELX=0.5) while the DELX in the trust region job is adaptively updated: between design cycles 2 and 6, DELX=1.0 and then reduced below 0.5 between design cycles 8 and 14. Finally, [Figure 9-2](#) plots the maximum constraint values with and without TR. [Listing 9-1](#) lists the summary of the design history.

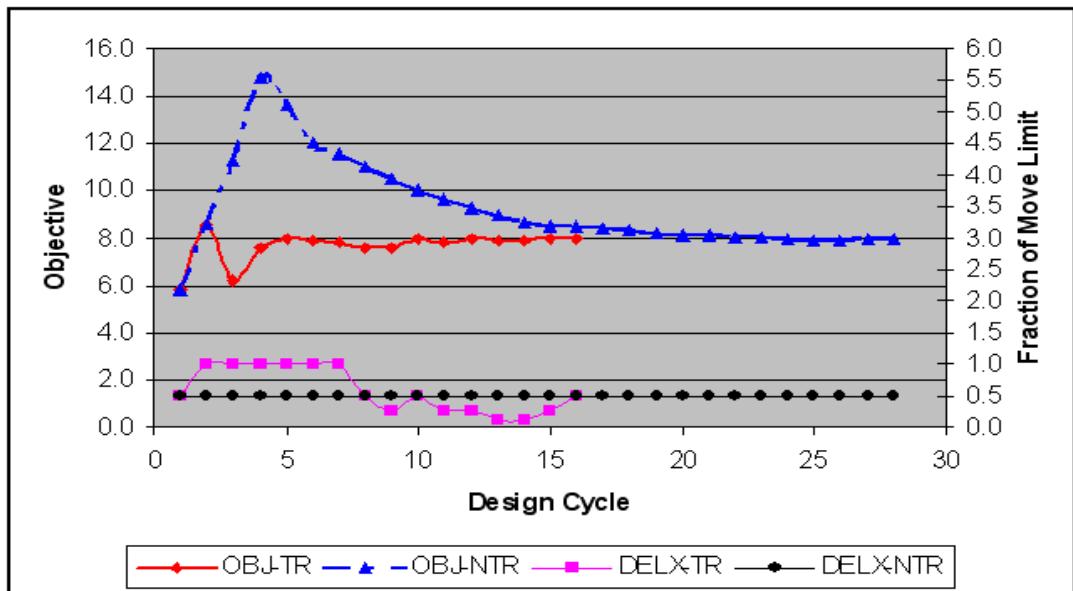


Figure 9-1 Objective History of Example 2: Trust Region vs. Non-Trust Region

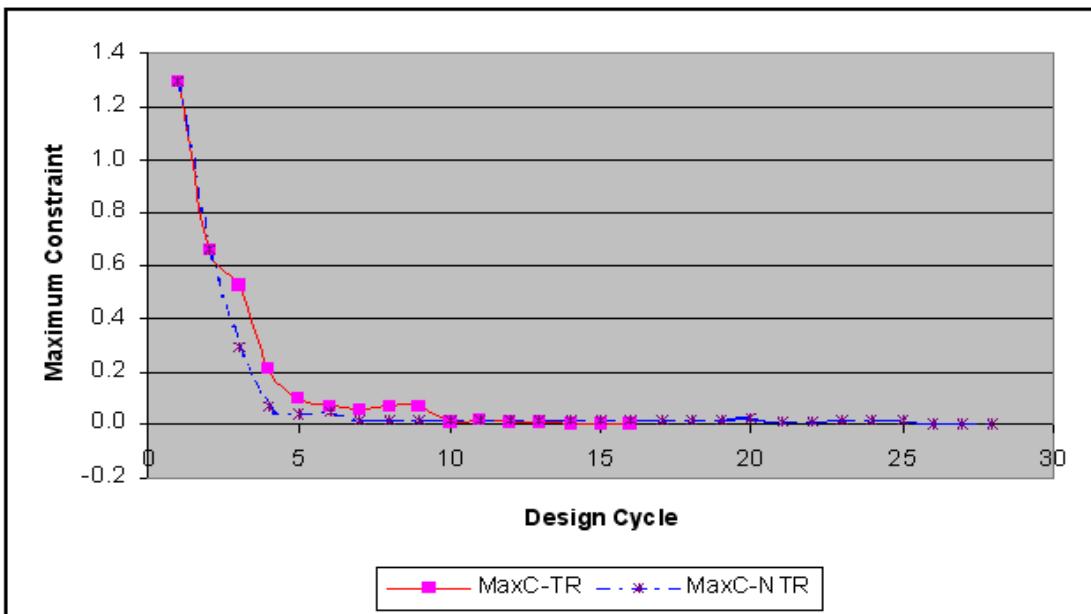


Figure 9-2 Max. Constraint History of Example 2: Trust Region vs. Non-Trust Region

Listing 9-1 Table Summary of Objective and Maximum Constraints

INITIAL				
1	8.610938E+00	5.784520E+00	8.610291E+00	7.509516E-05
2	6.194046E+00		6.193711E+00	5.412208E-05
3	7.550942E+00		7.550880E+00	8.146334E-06
4	7.982732E+00		7.982616E+00	1.451547E-05
5	7.912356E+00		7.912348E+00	1.084769E-06
6	7.772725E+00		7.772723E+00	1.840425E-07
7R	7.562489E+00		7.562516E+00	-3.657057E-06
8R	7.575960E+00		7.575985E+00	-3.398793E-06
9	7.943342E+00		7.943305E+00	4.622315E-06
10R	7.829101E+00		7.829069E+00	4.019795E-06
11	7.925030E+00		7.925027E+00	4.211797E-07
12	7.889203E+00		7.889194E+00	1.148394E-06
13	7.907179E+00		7.907176E+00	3.015218E-07
14	7.924466E+00		7.924470E+00	-5.415548E-07
15	7.924493E+00		7.924493E+00	0.000000E+00



Superelement Optimization

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.

MSC Nastran provides a variety of techniques for performing superelement analysis and you are referred to the MSC Nastran Superelement User's Guide for a discussion of these powerful techniques. For the purposes of this discussion it is necessary to identify three basic ways of incorporating superelements in your design optimization task.

1. List Superelements that are defined in the main bulk data section by specifying lists.
2. PART Superelements have a separate partitioned bulk data section that defines each superelement in a standalone fashion. The main data section residual information and acts to integrate the parts into a complete analysis.
3. External Superelements are represented in the run by matrices provided to the overall analysis.

A final concept is that of an image superelement, which allows you to duplicate one of the list or parts superelements.

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

Supported Superelements

List, parts and image superelements only may be used in connection with the design model. List, parts, image as well as external superelements may 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 just as in the case with non superelement models by using either linear relationships specified on DVxREL1 Bulk Data entries or user-defined equations on DVPxEL2 entries. Design variable-to-property relations may be local to a given superelement or may span superelement boundaries. For example, a design variable may be related to a particular property entry which is, in turn, used in a number of superelements in the model.

Each Part Superelement may contain DESVAR and DVxREL1/2 entries for that particular part on a standalone basis. The SEDLINK entry can be used to provide cross boundary support and must be in the residual bulk data section.

You can also define design variable-to-grid relationships using DVGRID Bulk Data entries. A given design variable may 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.

Each Part Superelement may contain DRESP1, DRESP2, DRESP3 entries for that particular part on a standalone basis. The SEDRSP2 and SEDRSP3 entries can be used to provide cross boundary support and must be in the residual bulk data section. If the weight or volume of a particular part is used as a response, the associated DRESP1 should be in the residual. If the objective is for an element or grid response that is in a part, it is necessary to point to a SEDRSP2 in the residual that in turn points to the DRESP1 in the part.

Design Responses in Superelement Design Modeling

You can select design responses using DRESP1 and/or DRESP2 Bulk Data 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 (SEID_i), 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 the residual structure, or Superelement 0. By default, only those elements belonging to the residual superelement are included in weight and volume computations. The design responses input to DRESP2 equations can span multiple superelements or load cases.

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 which span superelement boundaries, the screening criteria will apply individually to all superelements that share responses listed on this DRESP1 entry. 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) and Generalized Dynamic Reduction (GDR).

Punch of PARTS Design Results

For user convenience, specifying PARAM,[PSENPC](#),YES will write the updated bulk data entries into separate '.pch' files for each Part Superelement, and each design cycle. Note that the number of design cycles that are output is dependant on the value of PARAM,[DESPCH](#). The advantage of specifying PARAM,PSENPC,YES is that each of the '.pch' files with the updated design bulk data can be used to replace the original model with an 'INCLUDE' entry after the appropriate 'BEGIN SUPER=*seid*'. If PARAM,PSENPC,YES is not specified, the updated bulk data entries for all Part Superelements are written to a single '.pch' file which will require you to manually extract each Part Superelement model from the .pch file and place it in the appropriate 'BEGIN SUPER=*seid*' section of the Bulk Data Section.

PARAM,PSENPC	Default=NO. Setting PSENPC to YES causes updated bulk data entries of a Part Superelement for a design cycle punched to a separate file named as follows
JOBNAME_psexxx_yy.pch	Where xx is for Part Superelement ID and yy is for design cycle. Note that PARAM, PSENPC has no effect for non-Part Superelement run.

Output

For SOL 200 with design models in each Part Superelement, the .f06 output is similar to non Part Superelement optimization jobs. There are some minor differences that are specific to Part Superelements only.

The "Comparison Between Input Property Values from Analysis and Design Model" section of the .f06 will repeat for the design model of each Part Superelement. A sample for a Part Superelement is shown as follows:

DOUBLE FLYSWATTER MODEL \$ ANALYSIS USING PART SUPERELEMENTS S.E. STATICS - MULTIPLE LOADS	JANUARY 13, 2009 MSC Nastran 1/12/09 SUPERELEMENT 4							
----- COMPARISON BETWEEN INPUT PROPERTY VALUES FROM ANALYSIS AND DESIGN MODELS -----								
PROPERTY TYPE	PROPERTY ID	PROPERTY NAME	ANALYSIS VALUE	DESIGN VALUE	LOWER BOUND	UPPER BOUND	DIFFERENCE FLAG	SPAWNING FLAG
PSHELL	4	T	5.000000E-02	5.000000E-02	5.000000E-03	1.000000E+20	NONE	*
1. 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 AND THE DESIGN MODEL IS BEING USED TO OVERRIDE THE ANALYSIS MODEL. IF THE FLAG IS FATAL, THEN THE DIFFERENCES ARE GREATER THAN 1.000000E+35 AND THE RUN WILL BE TERMINATED. 2. THE SPAWNING FLAG (*) INDICATES THAT THE SPAWNED PROPERTY IS DERIVED EITHER FROM THE BEAM CROSS SECTION LIBRARY OR FROM A PBEAM ENTRY. THE PROPERTY ID FOR THE SPAWNED PROPERTY IS IDENTICAL TO ITS PARENT.								



Note that the Part Superelement ID shows up in the title line of a page. In addition, the residual Part Superelement has two versions of above output, first and last. If there are differences between design and analysis model for residual Part Superelement, the differences will show up in the first version.

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. Guyan reduction (the default) is exact for stiffness properties but only approximate for mass. Thus, you may want to consider using an advanced reduction method, such as component modes synthesis, to improve the sensitivity analysis accuracy.
- 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.
- Part Superelement optimization does not support topology (TOPVAR), topography (BEADVAR), or topometry (TOPMVAR) optimization.

Example Using Part Superelement

TPL Problem d200pse1.dat

The double-headed fly swatter model will be used to demonstrate Part Superelement Optimization with the design model SE 0 including DESVAR from each Part.



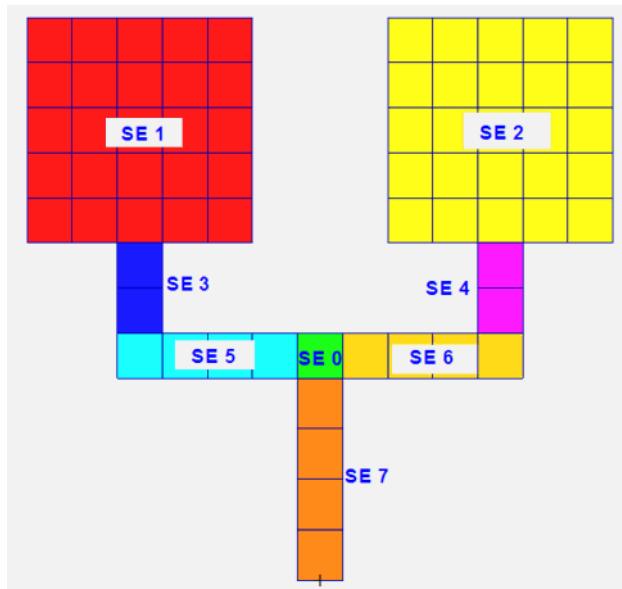


Figure 9-3 Example problem d200pse1

The output is similar to output from non Part Superelement Optimization runs, as an example, the Design Variable History for d200pse1.dat is:

Listing 9-2 Design Variable History Partitioned Superelements

DESIGN VARIABLE HISTORY								
INTERNAL DV. ID.	EXTERNAL DV. ID.	LABEL	INITIAL	1	2	3	4	5
1	110	T10	1.0000E+00	1.2861E+00	1.0289E+00	8.2310E-01	6.5432E-01	5.4130E-01
2	101	T1	1.0000E+00	1.2873E+00	1.0299E+00	8.2389E-01	6.2389E-01	4.2389E-01
3	1001	GE_1	1.0000E+00	1.0000E+00	1.0000E+00	1.0000E+00	1.0000E+00	1.0000E+00
4	102	T2	1.0000E+00	7.4711E-01	5.4711E-01	3.4711E-01	1.7355E-01	1.0000E-01
5	103	T3	1.0000E+00	1.3277E+00	1.3355E+00	1.0683E+00	1.2011E+00	1.1998E+00
6	104	T4	1.0000E+00	7.3786E-01	5.3786E-01	3.3786E-01	1.6893E-01	1.0617E-01
7	105	T5	1.0000E+00	1.3298E+00	1.5627E+00	1.2502E+00	1.2792E+00	1.3958E+00
8	106	T6	1.0000E+00	7.2738E-01	5.2738E-01	3.2738E-01	1.6369E-01	1.0000E-01
9	107	T7	1.0000E+00	1.3298E+00	1.5957E+00	1.8369E+00	1.9337E+00	2.0531E+00

DESVAR as defined in each PART. Duplicate IDs are allowed between Parts.



Automatic External Superelement Optimization (AESO)

Introduction

With superelements, you can manually partition the analysis model into two parts: a designed part and a non-designed part. The latter is treated as one external part superelement while the former is defined as a residual structure. A creation run is performed which applies Component Mode Synthesis (CMS) or Static Condensation to the part superelement and stores the resulting boundary matrices in a database or a punch file. The optimization task is performed on the assembly run that assembles those boundary matrices into the residual model for solving system solutions. The strategy is most efficient when the size of the design model (or residual model) is much smaller than the size of the original analysis model. However, although the feature is efficient in CPU time, since both files of both creation and assembly runs must be created by you, significant effort in manual partitioning the model might outweigh the performance gain.

The AESO capability presented here extends this manual process in an important way: rather than requiring you to segregate your large model into a designed and non-designed part, the process does this automatically by identifying which parts of the finite element model are affected by the design task. In essence, the new AESO capability provides an efficient and accurate solution in a user friendly way.

Benefits

Several major benefits are:

1. Removes a tedious and error prone task from the user in preparing the user input data.
2. Does not require you to be knowledgeable in the specialized area of superelements in general and external superelements in particular.
3. Provides an efficient and accurate approach for large-scale design optimization tasks.
4. Enable the performance of various design studies rapidly once the model has been divided into a designed and non-designed part. Examples of this are the setting up of different design constraints and objective in the studies to gain insight into the design and the available trade-offs or the applying of various frequency excitation loadings in the frequency response analyses.

Methodology

A complete AESO task involves two separate MSC Nastran job runs: 1) the first run is an AESO creation run (or simply creation run) whose logical flow is described in [Figure 9-4](#) and 2) the second run is an AESO assembly run (or simply assembly run) whose logical flow is described in [Figure 9-5](#).

As shown in [Figure 9-4](#), the creation run automatically partitions the original analysis model into the residual (the designed part) and external SE (the non-designed part). This automatic partition procedure will assign the following grid points to the residual:

1. All grid points that belong to a design model consisting of DRESP1, DVGRID, DVPRELi, DVMRELi and DVCRELi entries;



2. All grid points that are referenced on all static or dynamic loading entries such as DAREA, DPHASE, FORCE, MOMENT, PLOADi, TEMP entries;
3. All grid points for a rigid element that has one or more connecting grid belonging to the residual;
4. Any grid point that is a dependent grid on an MPC entry

After the automatic partition procedure, a new user input file is created from the residual (Figure 9-5). Then, the remainder of the creation run is the application of Static Condensation and/or Component Modes Synthesis procedures to produce stiffness, mass, damping boundary matrices. After the creation run is complete, a MSC Nastran database is saved to store the boundary matrices and a .asm file is also created to include superelement boundary connection information.

The assembly run is similar to a conventional SOL 200 task as shown in Figure 9-6 by utilizing all three types of data generated from the creation run (Figure 9-5). The original optimization problem is solved by assembling the boundary matrices into the residual for the system solutions.

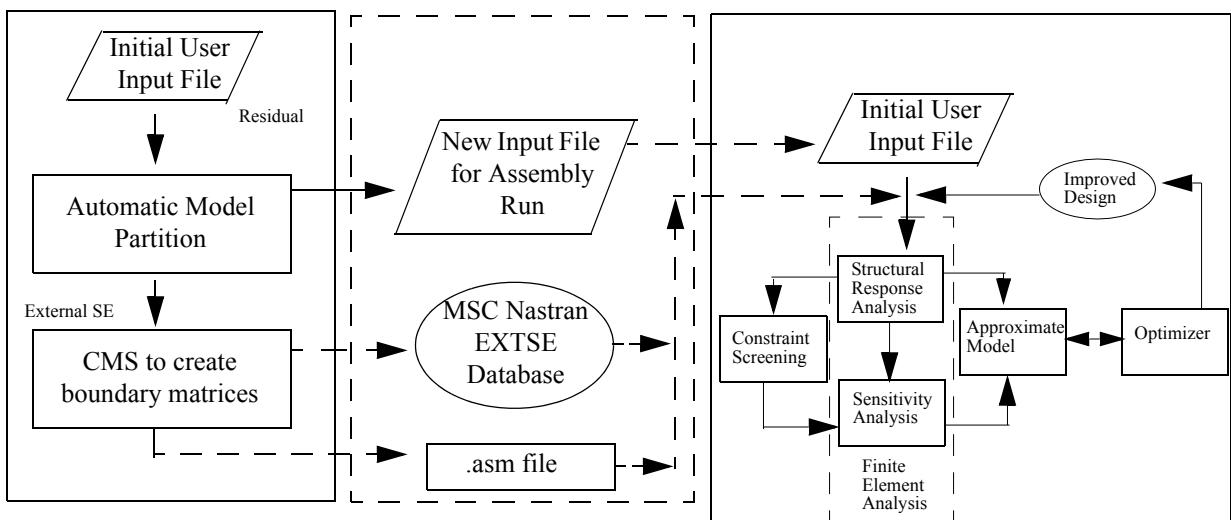


Figure 9-4

AESO Creation Run

Figure 9-5 Output of Creation Run and Input of Assembly Run

Figure 9-6

AESO Assembly Run

Input

1. The two parameters on the DOPTPRM entry are used to create an AESO job:
 - AUTOSE - flag to request an AESO job (integer 0, 1, Default = 0). AUTOSE = 1 activates an AESO creation run.
 - DRATIO - the threshold value that is used to turn off an active AESO job if the ratio of the size of the design model to that of the analysis model is greater than DRATIO (Real > 0.; Default = 0.1).



2. An ASSIGN statement with a logical key name 'AESO' is placed in the FMS to specify an input file name for the assembly run.

To illustrate ideas behind the input and output for an AESO task, a test problem (aesol.dat) is used here.

[Figure 9-7](#) shows a sample model whose upper left portion covering elements 18 to 42 (SE 1) is the non-designed part while the rest of structure is the residual structure.

The listing below is a condensed version of the creation run file (aesol.dat) that only shows the required user input to invoke an AESO creation run. The **assign aeso='aesol_2.dat'** statement is specified in the FMS section and **autose 1** and **dratio 0.9** are requested on the DOPTPRM entry. The DRATIO=0.9 here overrides the default.

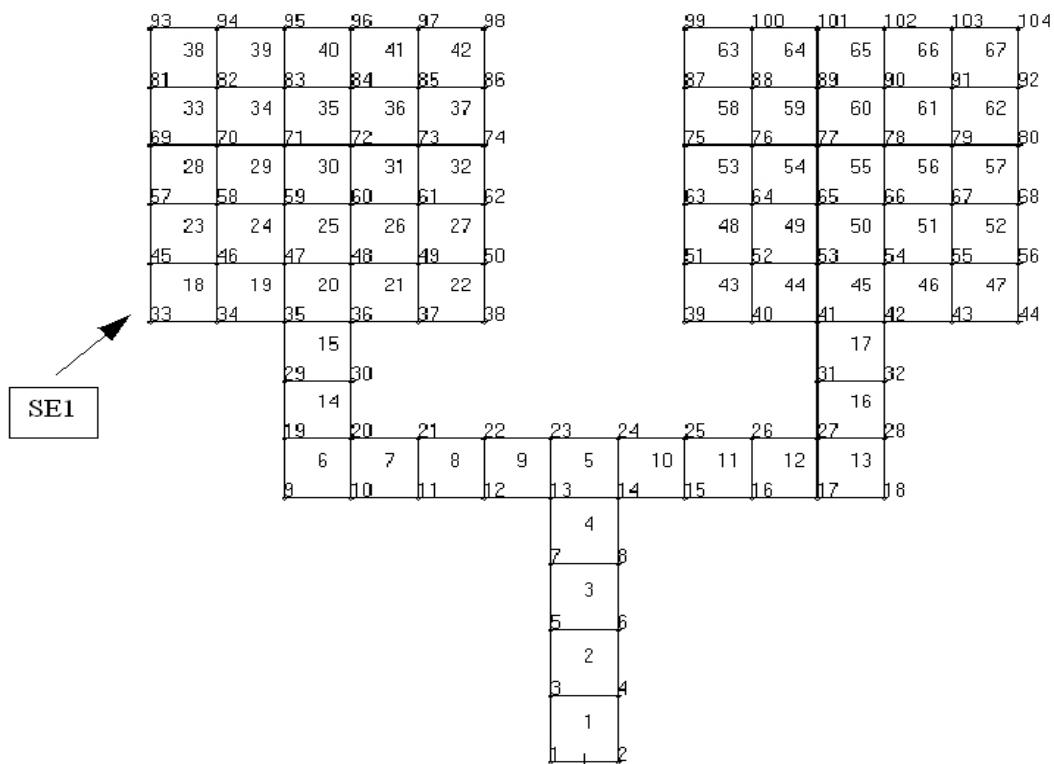


Figure 9-7 A Sample Model

```
assign aeso='aesol_2.dat'
SOL 200
CEND
desobj(max) = 1
analysis = modes
....
Begin Bulk
.....
doptprm desmax 10      autose 1          dratio 0.9
enddata
```

Figure 9-8 Condensed Version of the Creation Run File, dseoptl.dat

Outputs

As shown in [Figure 9-4](#), three types of data generated from the creation run are saved in the working directory: a Nastran database, a .asm file and a new input file for the assembly run (or an assembly file). This section describes each of these items and explains how they are used in the assembly run. In addition, some special print outputs from the creation run are shown that display the model partition information.

Nastran Database Files

Two Nastran database files: aesol.MASTER and aesol.DBALL are automatically saved when the creation run is submitted with SCR=NO option. Notice the size of the database is much smaller than the regular Nastran database because only boundary matrices are stored. The Master file will be referenced by an ASSIGN statement in the new input file described below.



The .asm File

The ‘aesol.asm’ file that includes the boundary connection information follows. As shown in [Figure 9-7](#), this problem has only two boundary points, 35 and 36 between the residual and the external superelement 1. This file is accessed through an INCLUDE command in the assembly file.

```
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
$ ASSEMBLY PUNCH (.ASM) FILE FOR EXTERNAL SUPERELEMENT 1
$ THIS FILE CONTAINING BULK DATA ENTRIES PERTAINING TO
$ EXTERNAL SUPERELEMENT 1 IS MEANT FOR INCLUSION
$ ANYWHERE IN THE MAIN BULK DATA PORTION OF THE ASSEMBLY RUN
$ SEBULK 1EXTERNAL MANUAL
$ SECONCT 1 0 5.0E-05
$ 35 35 36 36
$ BOUNDARY GRID DATA
$ GRID 35 -3.6 6. 0.
$ GRID 36 -2.8 6. 0.
$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$$
```

New Input File for the Assembly Run

Notice that the name of this file, aesol_2.dat is specified on the ASSIGN AESO statement in the creation run ([Figure 9-8](#)). It is a standard MSC Nastran input file. The AESO specific contents in the Executive Control, Case Control and Bulk Data Sections are listed in [Figure 9-9](#) and are described below.

```
nastran rseqcont=1
assign sel= './aesol.MASTER'
dblocate datablk(EXTDB) logical=sel,
CONVERT(SEID=1)
SOL 200
CEND

desobj(max) = 1
analysis = modes
.....
subcase 10
method = 1
spc = 10
$

begin bulk
include './aesol.asm'
.....
doptprm desmax 10
enddata
```

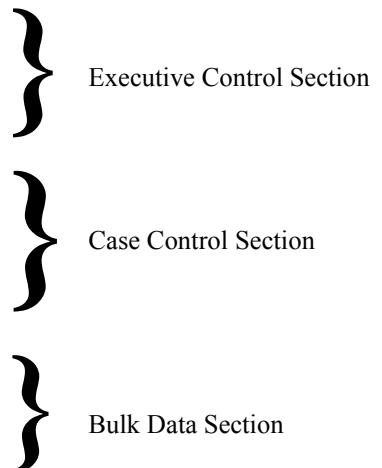


Figure 9-9 Highlights of the Assembly Run File, aesol_2.dat



1. Executive Control Section

The Nastran rseqcont=1 statement instructs the input file processor to ignore all continuation fields. This statement is automatically created in this file regardless of whether the creation run has it or not.

The next two statements assign the Nastran Master database file and locate the EXTDB datablock that stores various boundary matrices.

Notice that the other statements in Executive Control Section of the creation run are not retained.

2. Case Control Section

The whole Case Control section of the creation run is retained in the assembly file.

3. Bulk Data Section

This section completely defines the residual structure. The include './aes01.asm' command allows the assembly run to access the .asm file created from the creation run. In addition, the autose 1 and dratio 0.9 have been removed from the doptprm entry of the creation run.

Special Print Outputs from Creation Run

The following output is taken from the aes01.f06 file. It displays detailed information about the model partition. You may use [Figure 9-7](#) to help read the printout here. Notice that the Superelement 1 covers the non-designed part while the residual (or Superelement 0) covers the designed part.



Listing 9-3 Printout Showing Model Partition of Designed and Non-Designed Parts

BOUNDARY SEQUENCE ASSIGNMENT TABLE										
BOUNDARY SEQUENCE ID			ASSIGNED TO POINT ID (SUPERELEMENT)							
---		1B	35 (0)	36 (1)	35 (1)	SUPERELEMENT	0
									LIST OF INTERIOR POINTS	(TOTAL NO. OF INTERIOR POINT = 70)
INDEX	-1-	-2-	-3-	-4-	-5-	-6-	-7-	-8-	-9-	-10-
1	1	2	3	4	5	6	7	8	9	10
11	11	12	13	14	15	16	17	18	19	20
21	21	22	23	24	25	26	27	28	29	30
31	31	32	39	40	41	42	43	44	51	52
41	53	54	55	56	63	64	65	66	67	68
51	75	76	77	78	79	80	87	88	89	90
61	91	92	99	100	101	102	103	104	1B	2B
									SUPERELEMENT	0
INDEX	-1-	-2-	-3-	-4-	-5-	-6-	-7-	-8-	-9-	-10-
1	1	2	3	4	5	6	7	8	9	10
11	11	12	13	14	15	16	17	43	44	45
21	46	47	48	49	50	51	52	53	54	55
31	56	57	58	59	60	61	62	63	64	65
41	66	67								
									SUPERELEMENT	1
INDEX	-1-	-2-	-3-	-4-	-5-	-6-	-7-	-8-	-9-	-10-
1	1B	2B								
									SUPERELEMENT	1
INDEX	-1-	-2-	-3-	-4-	-5-	-6-	-7-	-8-	-9-	-10-
1	33	34	37	38	45	46	47	48	49	50
11	57	58	59	60	61	62	69	70	71	72
21	73	74	81	82	83	84	85	86	93	94
31	95	96	97	98						
									SUPERELEMENT	1
INDEX	-1-	-2-	-3-	-4-	-5-	-6-	-7-	-8-	-9-	-10-
1	18	19	20	21	22	23	24	25	26	27
11	28	29	30	31	32	33	34	35	36	37
		21		38		39		40		41

42

Guidelines and Limitations

- You may adjust DRATIO to allow an assembly run with larger or smaller residual model. The UIM 7824 provides brief information about the sizes of your analysis model and design model in terms of number of the grid points.

```
*** USER INFORMATION MESSAGE 7824 (DSGRDM)
THE NUMBER OF GRID POINTS IN THE ANALYSIS MODEL = 104.
THE NUMBER OF GRID POINTS IN THE DESIGN MODEL = 70.
THE DESIGN MODEL COMPRISSES 67.3 PERCENT OF THE ANALYSIS MODEL.
```

- For an AESO job with Analysis=MODES or MFREQ, it is recommended to activate the matrix domain based decomposition with ‘domainsolver acms(partopt=dof)’ in the Executive Control section to speed up the CMS procedure.
- Always specify the ASSIGN AESO=’filename.ext’ statement in the creation run to define the name of the assembly file. Directly assigning the original job name to filename should be avoided and will cause the assembly run to fail with User Fatal Message 713. A good practice is to add some suffix to the original file name such as myjob_2nd.dat where myjob is the original file name.



```
*** USER FATAL MESSAGE 732 (OPFUNT)
LOGICAL NAMES 'INPUT' AND 'AESO' ARE ASSIGNED TO THE SAME PHYSICAL
FILE.
USER INFORMATION: PHYSICAL FILE NAME 1: ./abc.aeso
PHYSICAL FILE NAME 2: ./abc.aeso
USER ACTION: CHANGE FILE NAME ON ONE OF THE ASSOCIATED ASSIGN STATEMENTS.
```

- When submitting the AESO creation run, use SCR=NO option. Otherwise, the Nastran database will not be retained after the creation run is done. However, it is optional for submitting an assembly run.
- After the creation run is complete, check the following user information message in the f06 file to ensure the job is terminated successfully.

```
^^^
^^^ USER INFORMATION MESSAGE 9181 (FEA)
^^^ THE JOB IS TERMINATED FOR AN AUTO EXTERNAL CREATION RUN
^^^
```

- The assembly input file may be modified to perform various design studies as long as the changes do not affect the boundary matrices stored in the database.
- If the AESO creation run includes a GRAV entry, it will be terminated with the following message. The same applies to TEMPD entry. They should be removed from the file if they are temporary inactive for the current task. Gravity and TEMPD loads are not supported with AESO since they apply loads to the entire structure and therefore block any partitioning.

```
*** USER FATAL MESSAGE 7699 (DSGRDM)
A GRAV Bulk Data entry is specified in an AESO creation run.
USER INFORMATION: The AESO run does not support the GRAV entry.
```

- Duplicate GRID entries are allowed in the .asm file and in the assembly input file. They will be automatically removed during the assembly run within the location tolerance specified by the TOL field on the SECONCT entry. For the AESO jobs, the default of the location tolerance has been increased to 5.E-5 from the original 1.E-5. However, due to numerical imperfection, this tolerance may need to be adjusted particularly for the cases in which the boundary grid points are defined in one or multi-levels of coordinate systems.
- Bulk Data parameter SEMAPPRT can be used to control the printout of the model partition information shown in [Listing 9-3](#). For example, Setting SEMAPPRT to 0 will turn off the printout.



- If the AESO task includes DVGRID entries, make sure that the grid points referenced by DVGRID entries are inside the residual. Since the grid points on DVGRID entries vary during the design process, including them as part of boundary grid points will invalidate the invariance of those boundary matrices. Currently, the grid points on the DVGRID entries will not be automatically assigned to be inside the residual. ASET and ASET1 entries can be used to create an enclosure or a barrier to ensure the grid points referenced on the DVGRID entries are always placed inside the residual.
- Since all the CORD1i entries are automatically converted to CORD2i entries during the AESO creation run. The DVGRID entry should not reference the grid points that define the CORD1i entry.
- The AESO tasks do not support acoustics response.



Randomization of a User's Input Data File

Introduction

The stochastic capability in MSC Nastran is a first step towards a complete and automatic self-randomization of a Finite Element model. The capability currently offers the user the possibility to automatically distribute tolerances and uncertainties with minimum effort. This reduces dramatically the burden on a user wishing to perform large-scale stochastic simulations. In fact, once the stochastic option is triggered, the entire Bulk Data Deck is randomized automatically and without further user intervention. The resulting model, which needs to be incorporated in a Monte Carlo Simulation loop - there are numerous off-the-shelf products which support this capability - possesses unprecedented levels of realism.

In order to make full use of this new development, it is necessary to resort to a multi-run environment, which can spawn a certain number of independent MSC Nastran executions, collect the results, and allow the user to perform statistical post-processing. With the self-randomization capability in MSC Nastran, all the user needs to define are the outputs he wishes to monitor, such as stresses, eigen-frequencies, temperatures, displacements, etc. There is no need to define inputs, as these are defined automatically by MSC Nastran.

Benefits

A basic assumption of MSC Nastran is that the inputs to the analysis are known exactly so that the computed responses are also known exactly. This is, of course, an invalid assumption in that there will always be some uncertainty in the inputs with a corresponding variations in the outputs. The MSC Nastran provides a way of introducing this uncertainty into the analysis process by automatically randomizing user input real numbers based on the input values and statistical quantities that characterize the variation.

Input

The randomization capability is driven by STOCHASTICS case control command. If STOCHASTICS=ALL is used, all real quantities on connectivity (those starting with C), Material and Property entries as well as any loads and SPCD quantities are modified based on a covariance factor of 0.05. A Gaussian distribution is used to randomly select the perturbed quantity with the restriction that the value can be no more than a specified number of standards deviations from the user input mean value. The default number of maximum standard deviations is 3.

Alternatively, the STOCHASTICS command can point to a STOCHAS bulk data entry that provides the ability to selectively randomize different types of input quantities using user specified covariance values and number of allowed standard deviations. In this case, only the types of input specified are randomized so that, for example, it is possible to randomize the loads input while leaving the property values unchanged.

Output

There is no new output produced by this capability at present.



Guidelines and Limitations

The randomization algorithm involves using a random number generator, a Gaussian distribution, the prescribed covariance and a mean value based on the user input to come up with a randomized value that is to be used in the analysis. In order to avoid physically meaningless properties, the random value is prescribed to be within m standard deviations of the input value, where m is a user input value with a default value of 3.0.

The product of $m * \text{COV}$ should not be greater than 1.0 to eliminate the possibility of the property changing sign.

The full benefit of this capability requires submitting multiple runs with the same randomization parameters. Each would produce a unique randomization and it is possible to collect the results of each of these analyses and produce statistical information on the variability of the responses. At present, MSC does not have software that performs these functions.

If the user input property value is 0.0, no randomization occurs. It is recommended that any property values that are 0.0 (say orientation angle on a PCOMP entry) be set to some non-zero value that is not negligible.



Optimization of Nonlinear Structural Responses (Pre-release)

Introduction

MSC Software's SOL 200 is a gradient-based multidisciplinary design optimization capability and has been widely used by clients in applying optimization techniques to linear structural analyses. Its success has led to the desire to extend these techniques to nonlinear structural analyses. Studies have been done to apply both gradient and non-gradient based approaches to the nonlinear structural analysis problems. The gradient approach involves design sensitivity analysis of nonlinear responses and mathematical programming. It provides accurate solutions but requires sensitivity calculations that are either too difficult in derivation, too expensive numerically or that become problematic due to the potential discontinuities in the responses as a function of design variables. Non-gradient based approaches often use Response Surface Methods to construct a surrogate model and the mathematical programming techniques are applied to the surrogate model. This approach is very general but is limited in the size of the design problems. An Equivalent Static Loads (ESL) based approach has been developed that transforms the original problem into an iterative solution of linear sub-optimization problems. The most attractive attribute of this approach is that it shares the best features in gradient and non-gradient based approaches and avoids the disadvantages of each approach. Therefore, it is able to solve small- or large-scale problems more efficiently. Furthermore, the approach can be implemented with the existing highly developed nonlinear analysis (e.g., SOL 400) and linear response optimization software systems (e.g., SOL 200). However, its limitation is that it may not support general design statement due to limited support of nonlinear response and element types and nonlinear analysis disciplines because it requires that any supporting nonlinear response type must have the equivalent response type in the linear system.

The nonlinear response optimization capability in MSC Nastran (ESLNRO) is based on the ESL concept and implemented within SOL 400. Only nonlinear displacement and stress responses are supported.

The capability is considered a research tool (as opposed to production tool) at this point due to the limitations in its capability and because it has seen limited use. Therefore, it is designated "pre-release" or "beta".

What is supported:

- Analysis = NLSTATIC,
- Geometry (large displacement), material and boundary nonlinearities,
- Nonlinear stress responses are available from the basic nonlinear elements: CBEAM, CROD, CTUBE, CQUAD4, CQUADR, CTRIA3, CTRIA6, CHEXA, CPENTA, CTETRA.
- Enhanced Nonlinear Elements (PBEMN1, PSNL1D, PSNL2D, PSLDN1) can be included in the analysis model as long as no TOPVAR is present. The stress responses of these elements may also be optimized if they can be mapped to that of the basic nonlinear elements.
- RTYPE = DISP, STRESS, WEIGHT, VOLUME, SPCFORCE, FRMASS, COMP.
- DRESP2
- DESVAR, DVPRELx, TOPVAR



What is not supported:

- TOMVAR, BEADVAR
- DVMRELx, DVCRELx
- CASI solver in linear response optimization for displacement and stress responses
- Both TOPVAR and enhanced Nonlinear Elements in the same job

Methodology

Basic Optimization Statement

A general nonlinear response optimization problem can be stated as below:

Find: X

Minimize: $F(X, U_{NL})$

Subject to: $g(X, U_{NL}) < 0$

$X_L < X < X_U$

and $K(X, U_{NL})U_{NL} = P$ and $\Phi(U) = 0$ for contact conditions

where $F(X, U_{NL})$ is the objective function such as structural weight, nonlinear compliance, maximum displacement or any user defined response, $g(X, U_{NL})$ is the design constraint such as displacement, stress, fractional mass in topology optimization or any user defined constraint. Since the finite element based nonlinear analysis solver is used, the basic equilibrium equation, $K(X, U_{NL})U_{NL} = P$ and the contact condition is any, $\Phi(U) = 0$ must also be satisfied. Notice the solution to the equilibrium equation requires iterative process in addition to the optimization design loop.



ESLNRO

The ESLNRO approach involves three-iterative-procedures:

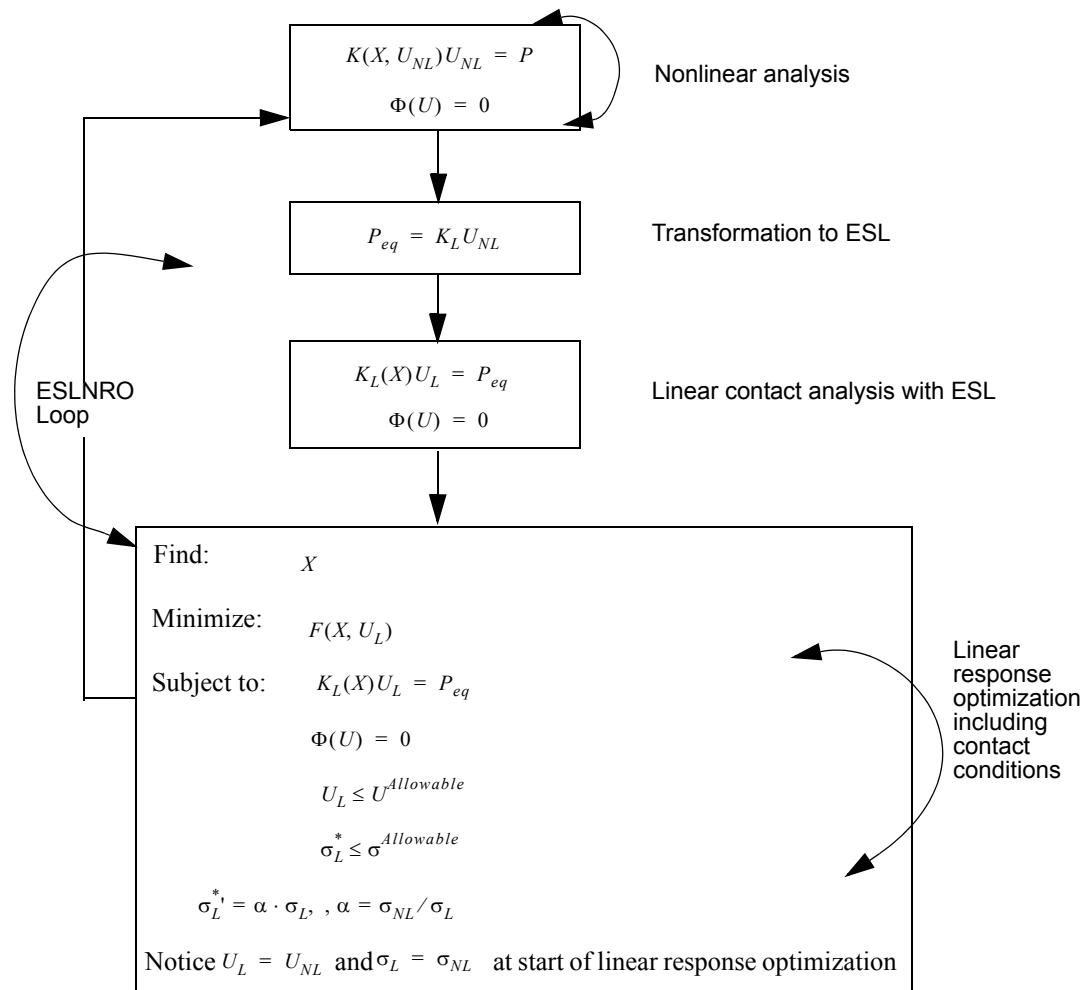


Figure 9-10

First, a nonlinear analysis is carried out. Next, the equivalent static loads (ESL) are computed from the nonlinear solutions. Then, the ESL is applied to a linear system and mathematical programming techniques are carried out on this linear system. The new design from the linear optimization is used to start a new ESLNRO loop. The process continues until the convergence criteria are satisfied. It is the ESL that establishes a platform to perform nonlinear response optimization without actually calculating the sensitivities of nonlinear responses.

One key ingredient in the ESLNRO is the generation of the equivalent static loads. According to Ref. 17., for a particular nonlinear response, a required ESL should produce an equivalent and identical linear



response at the start of the linear response optimization. The displacement-based ESL is computed by multiplying the linear stiffness matrix and nonlinear displacement solution and satisfies the requirement. For the stress-based ESL, Ref. 17. has used a more involved approach by solving an extra linear system with the nonlinear stress field as the initial condition without external loading. Then, the extra displacement solution is multiplied with the linear stiffness matrix to generate the stress-based ESL. Furthermore, a stress ratio scheme is introduced to ensure the linear stress filed will be identical to the nonlinear stress field. Notice that the ESLNRO in MD Nastran R3 directly uses the displacement-based ESL as the stress-based ESL to avoid the extra linear analysis. However, the stress ratio scheme is still applied to ensure that the linear stress responses are identical to the nonlinear stress response at the start of the linear response optimization.

As shown in [Figure 9-10](#), an ESL-based nonlinear response optimization task involves two types of loops. An inner loop (or a SOL 200 loop) is carried out in the linear response optimization and follows all the rules in a SOL 200 job. The ESLNRO loop is the outer loop that brings the nonlinear analysis and linear response optimization together. Like the inner loop carried out in SOL 200, the ESLNRO loop also has its own design move limit and the convergence criteria.

Design Move Limits in ESLNRO

In the ESLNRO, the actual nonlinear response optimization is solved by iterative solutions of linear sub-optimization problems. Although the linear responses at the beginning of the linear system optimization are identical to the nonlinear responses, there is no guarantee that the nonlinear responses evaluated with the proposed design are the same as those linear response evaluated with the same design. The design proposed by a linear sub-optimization solution may be too aggressive to affect convergence negatively. Ref. 17. has proposed a scaled-back scheme to limit the design move at each design cycle. Its main idea is to scale back the design move proposed by a linear sub-problem solution:

$$X_k^* = X_{k-1} + (X_{k-1}^1 - X_{k-1}) \cdot DELXESL$$

where X_k^* is the design variable for the k-th design cycle, X_{k-1} is the design variable at (k-1)th design cycle, X_{k-1}^1 is the proposed design from the linear optimization solution at (k-1)th design cycle and DELXESL is the fractional change allowed in each design variable during the ESLNRO loop.

An alternate to the scaled-back scheme is to limit the design move by posing more restrictive lower and upper bounds on each design variable. The following equations are used to update the design variable bounds. Subscript k indicates k-th design cycle, o indicates the initial design cycle, and i indicates i-th design variable, L lower bound and U upper bound. The initial design variable bounds are those specified on the DESVAR entries and DXMIN is a DOPTPRM parameter and is the same parameter used in a SOL 200 run.



$$X_k^L = \max(X_o^L, X_i - MOVE)$$

$$X_k^U = \min(X_o^U, X_i - MOVE)$$

$$MOVE = \max(DXMIN, abs(X_i) \cdot DELXESL)$$

It has been found that each scheme is effective in certain applications. Therefore, a user selection is provided.

The Convergence Criteria for ESLNRO

An ESLNRO job will be terminated if one of four conditions is satisfied. The design cycle here refers the ESLNRO design cycle (or the outer loop) not the design cycle in the linear response optimization (or the inner loop).

1. the maximum change among all design variables between the current and previous design cycles is less than a given tolerance (CONVDV) for two consecutive design cycles or
2. the linear response optimization task achieves hard convergence in a single inner design cycle for two consecutive ESLNRO design cycles or
3. the percentage of design variables that have their relative changes satisfy the tolerance (CONVDV) exceeds the user-supplied tolerance (TOPOCONV) for two consecutive design cycles. This only applies to topology optimization applications or
4. the maximum number of design cycles (DSMXESL) is reached

For an ESLNRO job terminated due to conditions 1 through 3, the following message will be printed out in the f06 file:

```
*****
***** ESLNRO CONVERGENCE ACHIEVED ON THE FOLLOWING CRITERIA *****
***** (HARD CONVERGENCE DECISION LOGIC) *****
1) MAXIMUM OF RELATIVE D.V. CHANGES = EEEEEEEEEE IS LESS THAN
   EEEEEEEEEE FOR 2 CONSECUTIVE ESLNRO DESIGN CYCLES;
   --- OR ---
2) TWO CONSECUTIVE LINEAR RESPONSE OPTIMIZATION RUNS ACHIEVED
   HARD CONVERENCE IN A SINGLE DESIGN CYCLE.
   --- OR ---
3) FFFFFFFF% OF DESIGNED ELEMENTS OR IIIIIII DESIGNED ELEMENTS HAVE
   THE RELATIVE VALUE CHANGE THAT IS LESS THAN THE EEEEEEEE (CONVDV) .
   THE HARD CONVERGENCE IS ARCHIEVED IF IT EXCEEDS FFFFFFFF% (TOPOCONV) .
*****
```

Figure 9-11 Special Strategy to Handle Divergent Nonlinear Analyses

During a topology optimization process, the program adds density to the element with higher strain energy and reduces density from the element with lower strain energy to minimize the compliance while maintaining the weight constraint. Iteratively, this will result in a collection of elements with lower strain



energy or ‘empty’ elements. Although creating empty and non-empty element groups is the goal of topology optimization, the empty elements introduce singular behaviors to the analysis itself. When enough ‘empty’ elements exist in the structure, the nonlinear analysis tends to fail to converge. In MD Nastran 2010, a simple strategy has been implemented to handle divergent nonlinear analyses. If a nonlinear analysis fails to converge at the initial design cycle, the job is terminated with appropriate message for the user to improve the model. However, if a divergent analysis is encountered for design cycles greater than 1, the design move produced by previous design cycle is cut by half and a new analysis is performed on the reduced design. This heuristic strategy will be consecutively for five design cycles. If the nonlinear analysis still fails to converge, the job will be terminated.

Implementation

The ESLNRO capability is implemented in MSC Nastran environment with nonlinear solver SOL 400, a comprehensive and sophisticated nonlinear solution sequence that can deal with general applications with geometric, material and boundary nonlinearities . The design logic for ESLNRO is shown in [Figure 9-10](#).

Note that only a single user input file is required that specifies the nonlinear analysis model as well as the design model with its design variables and constraints. However, internally, a multiple Nastran invocation strategy is used to bring SOL 400 and SOL 200 together to provide an integrated solution to the design task. Specifically, a dashed frame as shown in [Figure 9-10](#) forms the main ESLNRO loop in which the iterative solutions of linear sub-optimization problems are obtained through SOL 200 and SOL 400. The communication between the main driver, the SOL 400 and SOL 200 runs are established through various intermediate files. [Outputs](#) and [Guidelines and Limitations](#) will describe them and discuss how to manage these files.



Program Flow Chart

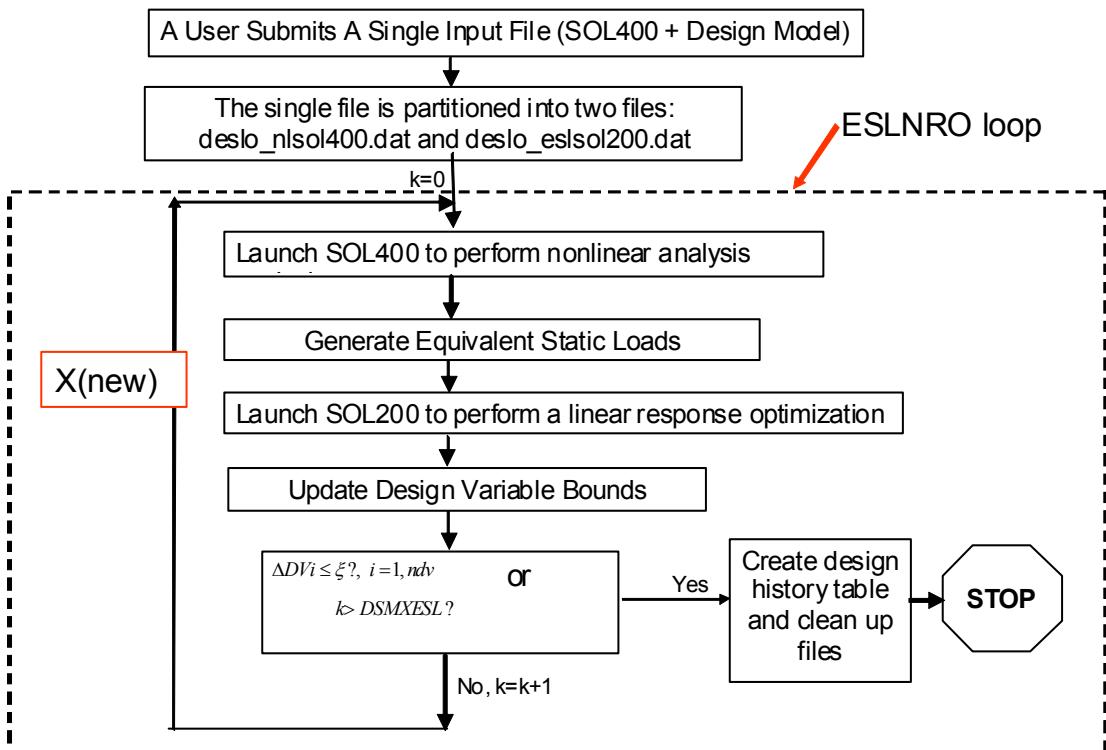


Figure 9-12 Program Flowchart

Input

In general, the required user input to perform an ESLNRO task is to add a design model definition to an existing SOL 400 job. The detailed description will be shown in [Examples](#). Here, several new types of input, which may be required to perform ESLNRO tasks are described.

1. Activation of ESLNRO

To invoke ESLNRO, you are required to specify a Nastran ESLOPT statement.

System Cell Name (Number)	Function and Reference
ESLOPT (443)	Flag to invoke ESLNRO concept of Equivalent Static Loads 0 – No ESLNRO, default 1 – Turn on ESLNRO



Example, to activate ESLNRO, use

Nastran ESLNRO = 1 or
Nastran system(443) = 1

2. Control parameters for ESLNRO tasks

New parameters, DELXESL and DSMXESL are added to the DOPTPRM entry. DELXESL is used to control how much a design variable can move during a ESLNRO design cycle while DSMXESL is the maximum allowable number of design cycles.

Name	Description, Type, and Default Value
DELXESL	Fractional change allowed in each design variable during the ESLNRO loop (Real > 0.0, Default = 0.5)
DSMXESL	Maximum number of design cycles applied to the ESLNRO loop (Integer ≥ 0 , Default = 20).

3. Definition of designed properties

Element property entries such as PBEAM, PROD, PSHELL and PTUBE can be specified on a DVPRELi entry. The associated nonlinear element types are: CBEAM(94), CONROD(92), CQUAD4(90), CQUADR(173), CROD(89), CTRIA3(88), CTRIAR(174), CTUBE(87). The property names on these entries that can be referenced on a DVPRELi entry shown in the following table:

Property Entry	Property
PBEAM	(A(i), I1(i), I2(i), I12(i), J(i), NSM(i), C1(i), C2(i), D1(I), D2(i), E1(i), E2(i), F1(i), F2(i), (i=A, B, 1 ... 9)), K1, K2, S1, S2, (NSI(j), CW(j), M1(j), M2(j), N1(j), N2(j), j=A, B)
PROD	A, J, C, NSM
PSHELL	T, 12I/T**3, TS/T, NSM, Z1, Z2 (The 12I/T**3 term can be designed but must be referenced by Field ID=6 rather than by name.)
PTUBE	OD, T, NSM

DVMRELi and DVCRELi entries are not supported.

4. New input for defining nonlinear responses with a DRESP1 Bulk Data entry

The displacement response is identified on a DRESP1 by RTYPE=DISP while the stress response is identified by RTYPE=STRESS. The linear displacement response on a DRESP1 can be used to define a nonlinear displacement response. However, defining nonlinear stress response requires specifying a nonlinear stress item code on the ATTA field of a DRESP1 entry. These stress item codes can be found in [Element Stress \(or Strain\) Item Codes](#) in the *MSC Nastran Quick Reference Guide*.



Stress responses from the following nonlinear elements are supported: BEAM(94), QUAD4(90), TRIA3(88), QUADR(172), TRIAR(173), HEXA(93), PENTA(91), TETRA(85). In order to ensure to support the nonlinear stress responses that have the equivalent linear stress responses, the nonlinear element stresses are categorized into three groups:

- a. the stress having the same name and same meaning as those in the linear element stresses;
- b. the stress having the different name but having the same meaning; and
- c. the stress having the different name and different meaning.

Only the stresses in groups-a and-b can be specified on a DRESP1 entry. The following lists the stresses from Groups-b and -c for supported nonlinear elements. For example, the Equivalent Stress is a group-b stress because it is equivalent to the von Mises stress in a linear element although their names are different. However, total strain, effective plastic strain and effective creep strain (as shown in ***bold italic***) cannot be specified on a DRESP1 entry because they do not have linear equivalents.

Nonlinear 2D

QUAD4 (90) (equivalent stress) (*effective plastic strain, effective creep strain*)

TRIA3(88) (equivalent stress) (*effective plastic strain, effective creep strain*)

QUADR(172) (equivalent stress) (*effective plastic strain, effective creep strain*)

TRIAR(173) (equivalent stress) (*effective plastic strain, effective creep strain*)

Nonlinear 3D

Hexa (93) (effective stress), (*effective plastic strain, effective creep strain*)

Penta (91) (effective stress), (*effective plastic strain, effective creep strain*)

Tetra (85) (effective stress), (*effective plastic strain, effective creep strain*)

5. ESL Bulk Data Parameters

Option to save ESLNRO intermediate files on disk

PARAM,ESLFSAV,character string (character, Default = NO)

ESLFSAV = YES requests that all the intermediate files from an ESLNRO job be saved on disk. The destination of these files can be directed with the 'sdir=' option on a Nastran submittal command line.

Selection of move limit schemes

PARAM,ESLMOVE,Integer, Default = 0

ESLMOVE = 0 selects a move limit scheme that poses restrict lower and upper bounds on design variables during the linear response optimization. ESLMOVE = 1 selects a move limit scheme that scales back the design move proposed from a linear response optimization.

User-supplied RC file

PARAM,ESLRCF,filename (Char*8, must be lower case). Default = blank

New Bulk Data parameter entry, PARAM,ESLRCF,filename allows the user-supplied RC file for the internally spawned jobs where filename is a character string up to 8 characters. Only lower case is supported.



Example:

PARAM,ESLRCF,*myrc* where *myrc* is the name of the user-supplied RC file with the following contents:

```
MEM=200m
EXE=~local_path/MSCNASTRAN
DEL=~local_path/SSS
```

The example shows a user-supplied RC file that requests each spawned SOL 200 or SOL 400 job be run with memory allocation of 200 million words per run and with executable and delivery database.

ESLLCOMP

```
Default = NO
```

ESLLCOMP selects types of compliance response to be included in the design task. The nonlinear compliance response is defined using a DRESP1 entry with RTYPE=COMP for the ESLNRO topology optimization tasks. As the default, it is computed by the product of the applied nonlinear loads and corresponding nonlinear displacement components. Alternatively, ESLLCOMP=YES selects a linear compliance response that computed as the total work done by the equivalent static loads on the linear system.

ESLMPC1

```
Default = 0
```

This parameter applies only to the ESLNRO jobs with 3D contact. Its default has different meanings depending on the type of contact applications. As the default, for a glued contact ESLNRO job, a linear response optimization task will include a set of MPC entries that are created from the nonlinear analysis. For a touching contact ESLNRO job, the linear response optimization task will not include the MPC entries by default. Setting ESLMPC1 to a positive number will turn on the MPC inclusion.

ESLMPC1 = 1: uses the MPC entries created from the nonlinear analysis at the converged nonlinear analysis.

ESLMPC1 = 2 uses the MPC entries created at the beginning of the very first nonlinear analysis

ESLOPTEX

```
Default = 0
```

The parameter allows the user to perform an ESLNRO job at a targeted exit point. The allowable values of ESLOPTX are listed below with their description.

- 0 - Do not exit. Proceed with ESLNRO nonlinear response optimization.
- 1 - Exit after the initialization of the analysis and design model but before nonlinear FE analysis begins.
- 2 - Exit after nonlinear FE analysis ends.
- 3 - Exit after design constraint evaluation and screening.

ESLPRT



Default = 0

ESLPRT specifies how often the ESLNRO results are printed in the f06 file and saved in the xdb file. By default, the program will print the results to the f06 file at the first and the last design cycles and save the results to xdb (or op2) at the first and last design cycles on the disk (See ESLPRT1 for selection of result contents).

ESLPRT > 0, then the results are printed at the first design cycle; at every design cycle that is a multiplier of ESLPRT; and the last design cycle.

ESLPRT < 0, the no results are printed and saved.

ESLPRT1

Default = 7

ESLPRT1 specifies what type of results to be written to the f06 and to xdb (or op2). It may take any of the following base values or to a combination of these base values.

ESLPRT1 = 0, write no data.

ESLPRT1 = 1, write the nonlinear analysis results to the f06 file.

ESLPRT1 = 2, write the optimization data controlled by P1 and P2 to the f06 file.

ESLPRT1 = 4, save the nonlinear analysis results to the xdb (or op2) file.

ESLPRT1 = 8, save the linear response optimization results to the xdb (or op2) file.

For example, by default, results from the nonlinear analysis, the optimization data will be written to the f06 file and result data will be written to xdb or op2.

ESLUNT1, ESLUNT2

Default = 53, 54

File units are used to store and retrieve design variables and design properties for ESLNRO topology optimization tasks.

TOPOCONV

Default = 80

Parameter TOPOCONV is applicable only to ESLNRO topology optimization tasks. It sets a lower bound for the percentage of the design variables whose maximum relative changes are within the tolerance specified by CONVDV on the DOPTPRM entry.

By default, when more than 80% of the design variables show their maximum relative changes are within CONVDV, the job will be terminated.

Outputs

During the ESLNRO job, in addition to the primary Nastran result files (e.g., .f06, .f04 and log), files are generated internally for communications between the main driver and nonlinear analyses (SOL 400 run) and linear response optimizations (SOL 200 runs). These are temporary files and will be removed at the



job's completion by default. However, the user can use PARAM,ESLFSAV,YES to save them on the disk if necessary.

These two types of files will be described using a user input file named deslo.dat.

The Primary Nastran Result Files (deslo.f06, .f04, log, etc.)

These are regular output files from a Nastran job and follow the MSC Nastran naming conventions such as .f04, .f06 and log files. The .f06 file contains certain messages that are unique to an ESLNRO job. For example, the following information messages are printed in the .f06 file for each design cycle to provide a brief description of the ESLNRO process:

```
*****
*   E S L N R O   D E S I G N   C Y C L E = 11 *
*****
^^^ A NONLINEAR ANALYSIS JOB INITIATED WITH FOLLOWING COMMAND:
/nast/MSC20071t1/linux64/nastran /scratch./deslo_nlsol400 scr=yes bat=no rcf=my.rc out=/scratch./deslo_nlsol400
^^^ A NONLINEAR ANALYSIS JOB FOR THE ESLNRO COMPLETED.

^^^ A LINEAR OPTIMIZATION JOB INITIATED WITH FOLLOWING COMMAND:
/nast/MSC20071t1/linux64/nastran /scratch./deslo_eslsol200 scr=yes bat=no rcf=my.rc
out=/scratch./deslo_eslsol200
^^^ A LINEAR OPTIMIZATION JOB FOR THE ESLNRO COMPLETED.

^^^ NO HARD CONVERGENCE IS ACHIEVED IN THE ESLNRO LOOP. JOB CONTINUES
```

If a nonlinear analysis job is unable to converge and is terminated at design cycle 11 in the ESLNRO loop, the following User Information Message 6464 will be printed out in the deslo.f06 file. In addition, the deslo.f06 will also include the additional information on the lack of convergence is printed in the regular SOL 400 .f06 file (not shown).

```
*** USER INFORMATION MESSAGE 6464 (DELSOPT)
RUN TERMINATED DUE TO NONLINEAR ANALYSIS JOB UNABLE TO CONVERGE AT DESIGN CYCLE = 11.
```

In addition, for initial design cycle and final design cycle, the results from nonlinear analysis tasks and the optimization output data controlled by P1 and P2 on the DOPTPRM entry are always printed out in the .f06 file. However, no results output are printed in the .f06 file for the intermediate design cycles.

At the end of the design cycle, a summary of design cycle history and design variable history are printed in the .f06 file. If you are familiar with a SOL 200 job, they look very much like the design history tables from an SOL 200 task. Here is the sample printout of the summary of design cycle history.



SUMMARY OF DESIGN CYCLE HISTORY				
(HARD CONVERGENCE ACHIEVED)				
	NUMBER OF NONLINEAR FINITE ELEMENT ANALYSES COMPLETED	38		
	NUMBER OF OPTIMIZATIONS W.R.T. LINEAR MODELS	37		
OBJECTIVE AND MAXIMUM CONSTRAINT HISTORY				
-----	-----	-----	-----	-----
CYCLE NUMBER	OBJECTIVE FROM LINEAR MODEL OPTIMIZATION	OBJECTIVE FROM EXACT ANALYSIS	FRACTIONAL ERROR OF LINEAR MODEL	MAXIMUM VALUE OF CONSTRAINT
INITIAL 1	2.630691E-01 3.190933E-01	2.742739E-01	1.634111E-01	4.676274E-01 1.778786E-01

The Internally Spawning Files

Two user input files, one for the SOL 400 run and one for the SOL 200 run are internally generated derived from the primary user input file: deslo_nlsub.dat and deslo_eslsub.dat. deslo is the name of the primary user input file and _nlsub and _eslsub are suffixes to distinguish a SOL 400 job from a SOL 200 job. Each has a unique Executive Control Section and a Case Control Section. Each file also has unique Bulk Data entries and shares a portion of common Bulk Data entries.

Detailed descriptions of two user input files are as follows. Notice multiple INCLUDE entries are used to facilitate sharing common Bulk Data entries among two jobs, updating DESVAR entries or GRID entries for shape optimization at the end of each design cycle without the need to changing the actual input files.

Description of a SOL 400 Input File (deslo_nlsub.dat)

```
SOL 400
CEND
include '$$dir/deslo_nlsub.cas' $ = original subcase contents minus DESOBJ/DESSUB/DRSPAN
BEGIN BULK
include '$$dir/deslo_grid.blk' $ = all GRID entries. Original entries for initial design cycle and updated
entries for design cycle>1.
include '$$dir/deslo_desmod.blk' $ = all design model Bulk Data entries except DESVAR entries
include '$$dir/deslo_desvar.blk' $ = all DESVAR entries
include '$$dir/deslo_nlmat.blk' $ = nonlinear material entries such as MATS1, MATEP,MATF,
NLPARM
include '$$dir/deslo_loads.blk' $ = the original loading Bulk Data entries.
include '$$dir/deslo_model' $ = the remaining portion of the original Bulk Data entries
ENDDATA
```

Description of a SOL 200 Input File (deslo_eslsub.dat)

```
SOL 200
CEND
include '$$dir/deslo_eslsub.cas' $ = ESL Subcases + DESOBJ/DESSUB/DRSPAN
BEGIN BULK
include '$$dir/deslo_grid.blk' $ = all GRID entries. Original entries for initial design cycle and updated
```



entries for design cycle>1.

include '\$ssdir/deslo_desmod.blk' \$ = all design model entries except DESVARs

include '\$ssdir/deslo_desvar.blk' \$ = all DESVARs entries

include '\$ssdir/deslo_esl' \$ = Equivalent Static Loads Bulk Data entries

include '\$ssdir/deslo_model' \$ = the remaining portion of the original bulk data entries

All the intermediate files are stored in the same Nastran scratch directory defined by environment variable \$ssdir. It could be reset on the Nastran command line with sdir=local-path-directory. If SCR option is set to Yes, they will be removed from the directory after the ESLNRO job is complete. If SCR is set to No, they will be saved in the directory.

Descriptions of Individual Include Files

deslo_nlsub.cas	This file contains the original contents of the Case Control Section used by a SOL 400 run.
deslo_eslsub.cas	This file defines SUBCASES that reference load cases corresponding to the same number of Equivalent Static Loads. It is generated by the SOL 400 run and will be used by a SOL 200 job.
deslo_desmod.blk	This file contains all the design entries except DESVAR entries and is used by both a SOL 200 job and a SOL 400 job.
deslo_desvar.blk	This file contains all the initial DESVAR entries for the initial design cycle. For design cycle > 1, it contains updated DESVAR entries. The file is used by both SOL 200 and SOL 400 runs.
deslo_grid.blk	The file contains all the initial GRID entries at the initial design cycle. For shape optimization it contains the updated GRID entries for design cycle > 1. It is used by both jobs.
deslo_loads.blk	The file contains the loads Bulk Data entries from the original user input file. It is only used by a SOL 400 job.
deslo_esl.blk	The file contains the loads Bulk Data entries for the Equivalent Static Loads. It is generated by a SOL 400 run at every design cycle and is only used by a SOL 200 job.
deslo_nlmat.blk	The file contains nonlinear analysis specific data such as nonlinear material entries that are supported by this project such as MATEP, MATF, MATS1 and NLPARM and is used only by SOL 400 job.
deslo_model	This file contains the remaining Bulk Data entries after the entries in deslo_grid.blk, deslo_nlmat.blk and deslo_desmod.blk, deslo_desvar, deslo_loads are excluded from the original Bulk Data Section and is used by both jobs.

Guidelines and Limitations

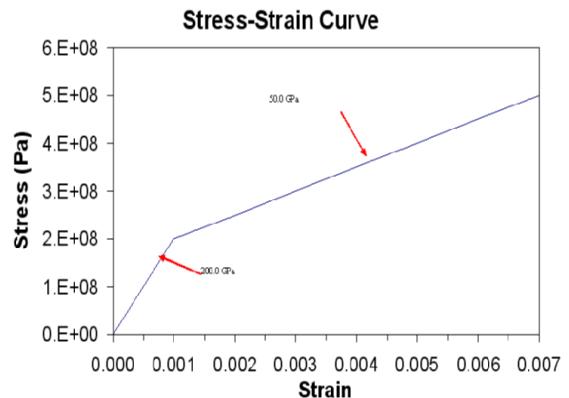
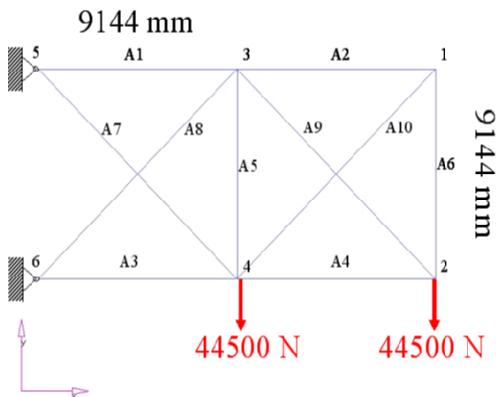
- To invoke the ESLNRO capability, set NASTRAN ESLOPT = 1 at the top of your input file.



- When you start an ESLNRO job, make sure the directory that will store the intermediate files does not contain any intermediate files from the previous ESLNRO run.
- You can specify your own RC file for these internally spawned SOL 200 or SOL 400 jobs using PARAM,ESLRCF, *RC_File_Name* to allocate more memory or for other purposes.
- After an ESLNRO job is complete, a complete Bulk Data Section with updated element properties entry or GRID and DESVAR entries for the last design cycle will be saved in the PCH file. In addition, the history of design objective, maximum constraints and design variables are also saved in the PCH file from which XY-Plots can be generated using spreadsheet program such as Microsoft Excel.
- If DRSPAN is used, PARAM, ESLLCOMP, YES must be included in your job for a job with a compliance response (i.e., RTYPE=COMP on a DRESP1 entry).
- Use smaller design move limits (both delxesl and delx) for the highly nonlinear ESLNRO tasks such as topology optimization with large deformation.
- For the design tasks that have a contact body relying on other contact body to prevent rigid body motion, the MPC equations should be included in the linear optimization using PARAM, ESLMPC1, 1. However, this setting does not produce good results as expected and requires more studies.

Examples

Example 1 10 Bar Truss (test library problem: deslo.dat)



(cross sectional areas)

Find:

Minimize: Weight

$$|\sigma_j| \leq 220 MPa \quad (j = 1, \dots, 10)$$
$$|\delta_{all}| \leq 100.0 mm \quad (\text{both x and y directions of all nodes})$$
$$78.5 mm^2 \leq X_i \leq 2826.0 mm^2 \quad (i = 1, \dots, 10)$$

This example demonstrates an ESLNRO optimization problem involving both geometric and material nonlinear behavior. The design task is to minimize the structural weight while maintaining nonlinear nodal displacements and element stresses within allowable limits. The job is terminated due to hard convergence to a feasible design. The following data compare the results between the initial design and the final design. Although both initial nonlinear displacement and stress constraints are violated, the final design is a feasible design.

- Max Deflection: Optimized: -99.85, Initial -146.76 (lower limit is -100.0)
- Max Axial Stress: Optimized: -218.26, Initial -283.99 (upper limit is 220.00)

Figure 9-13 shows the design history where an ESLNRO design cycle represents a nonlinear analysis followed by a linear optimization task. Each linear optimization task typically has its own series of design cycles as in a standard SOL 200 run. The blue line is for weight while the red line is for the maximum constraint. It is seen that a feasible design is attained after 10 design cycles but that the weight continues to decrease so that ultimately 37 design cycles are performed. Even for this small problem, it should be obvious that the number of nonlinear analyses required to solve the problem are much fewer than would be required if a response surface method approach had been used.



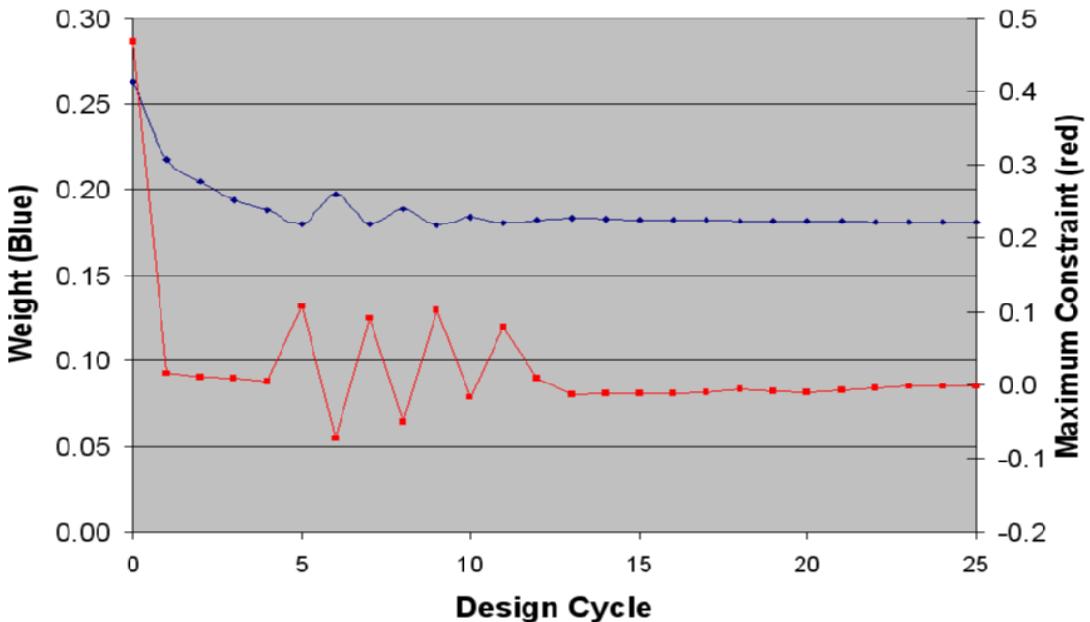


Figure 9-13 History of 10 Bar Truss Design

Example 2 Topology Optimization Problem with Large Deformation

(deslo9.dat, deslo9a.dat, deslo10.dat, deslo10a.dat and deslo9l.dat, deslo9al.dat)

This problem, originally published in Ref. 18., shows the necessity to include nonlinear analysis with topology optimization when large deformation is involved. A rectangular plate is modeled with CQUAD4 elements (not shown) and has the dimension of 0.8m by 0.2m by 0.001m. The middle points at both ends are fixed except the z-rotational degree of freedom. A concentrated force is applied at the middle point on the top as shown in [Figure 9-14](#).



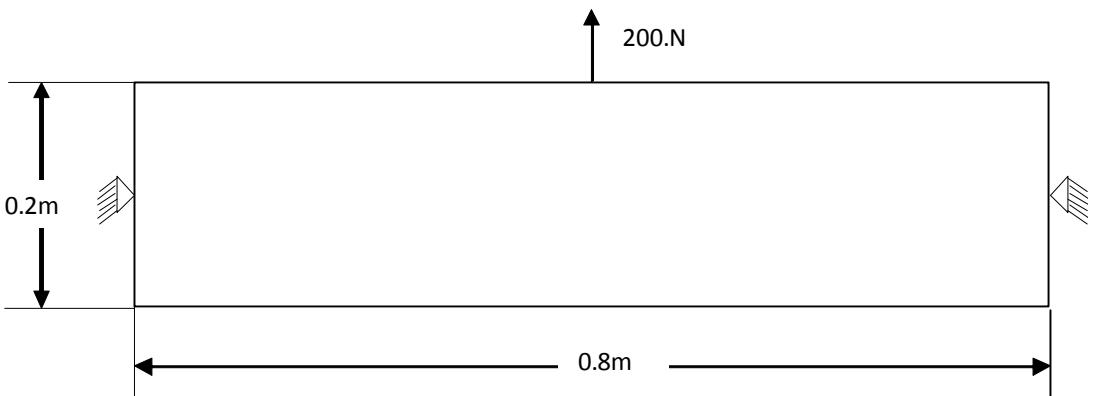


Figure 9-14 A Rectangular Plate Model

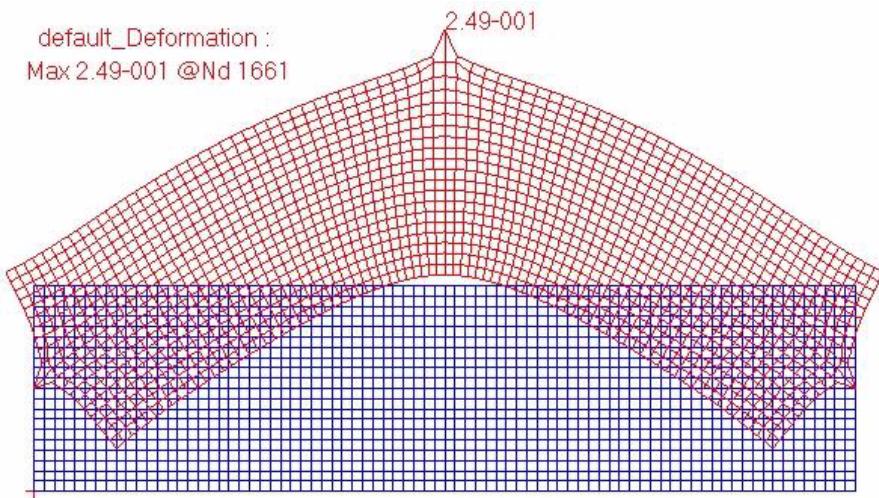


Figure 9-15 initial Deformation Plot

The deformation of the initial model is plotted as shown in [Figure 9-15](#). The maximum displacement occurs at the loading point with the value of is 0.249m, about 2.5 times of the height of the plate. So this is a nonlinear problem with a large deformation behavior.

The design task is to find an ‘optimal’ distribution of materials on this rectangular plate to minimize the nonlinear compliance while maintaining 20% of the original weight under the large displacement condition.



First, a linear topology optimization job of the same model is solved with SOL 200 (deslo9l.dat). It converges in 39 design cycles. The final configuration with is shown in [Figure 9-16](#). The plot is created using Patran's DesignTool/Post-Process with the threshold value of 0.2. Let us call it the updated linear structure (or FE model). One can see that the right and left arms of the updated linear structure are in compression under the original loading condition.

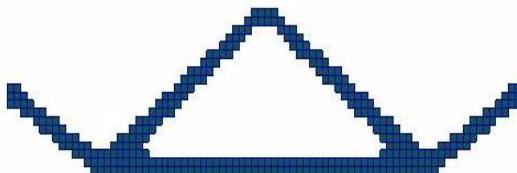


Figure 9-16 Final configuration from linear topology optimization

Next, the same job is solved by the new ESLNRO capability under the large displacement condition (deslo9.dat).

The job converges in 89 design cycles. Its final configuration is shown in [Figure 9-17](#) that is created using Patran's Design Tool/Post-Process/Display Results with the threshold value of 0.2. We will call it the updated nonlinear structure (or FE model). Notice the updated structure is in always tension.

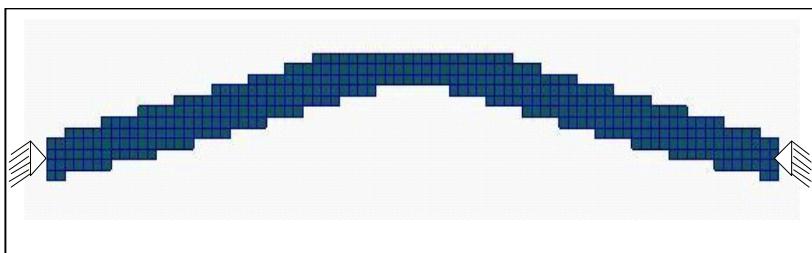


Figure 9-17 Final configuration from nonlinear topology optimization

It is clear that the updated linear and nonlinear structures are very different. One may notice that under the current loading condition (see [Figure 9-14](#)), a part of the updated linear structure is under compression while the updated nonlinear structure is all in the tension. To verify which structure is more capable of sustaining the nonlinear load condition, Patran's Group techniques are used to create an updated FE model for both the linear and nonlinear structures as shown in [Figure 9-16](#) and [Figure 9-17](#). Next, the nonlinear analysis is performed on each of the two structures with the original loading (deslo9al.dat and deslo9a.dat).

The final results are very enlightening: the updated linear FE structure collapses due to local buckling because the right and left arms are in compression. On the other hand, the updated nonlinear FE model performs well. [Figure 9-18](#) plots the deformation from the updated nonlinear FE model under the original loading. The maximum displacement of 0.0184m occurs at the loading point.



Patran 200XiX Development Only 01-Jul-09 17:33:29
Deform: SC1:DEFAULT, A1:Non-linear: 100, % of Load, Displacements, Translational,

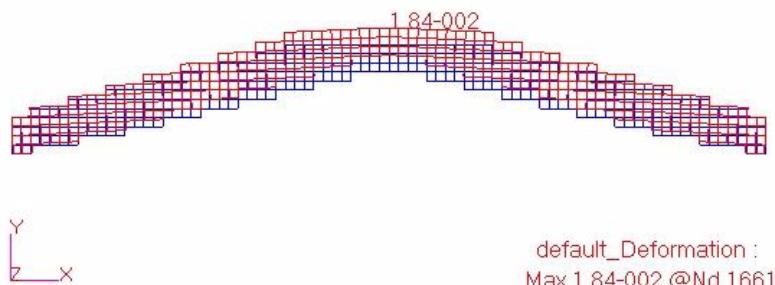


Figure 9-18 Displacement plot of the updated nonlinear structure

This example shows that it is necessary to perform the topology optimization by including nonlinear analysis responses.

Notice Ref. 18. published a final configuration that is given in [Figure 9-19](#). It is significantly different from both the updated linear structure and the updated nonlinear structure as shown in [Figure 9-16](#) and [Figure 9-17](#). According to Ref. 17., this updated nonlinear structure also performs well under the original loading.

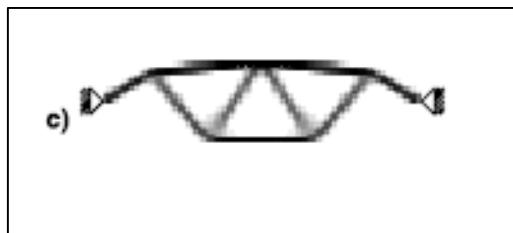


Figure 9-19 Final configuration obtained by Ref. 18.

It is natural to ask why it is significantly different from the updated nonlinear structure obtained by ESLNRO ([Figure 9-17](#)). One explanation lies in that the objective function used by Ref. 18. is nonlinear mean compliance that is the sum of nonlinear strain energy and complementary energy. However, the nonlinear compliance response in the ESLNRO is the sum of the product of applied loads and the corresponding nonlinear displacement components or the work done. In the linear analysis, the strain energy and its complementary energy are same. So the total work done is twice the strain energy.

However, in the geometrically nonlinear analysis, the above relation does not hold. The nonlinear compliance response defined in ESLNRO is different from the mean compliance response.

To carry out the study further, a new ESLNRO job is performed by setting DESMAX to 1. This enforces that the linear optimization job is only carried out in one design cycle and it then immediately returns the



proposed new design back to the ESLNRO loop. A new set of equivalent static loads will be generated from the updated nonlinear analysis. The purpose of this exercise is to update the equivalent static loads as much as possible since the equivalent static loads in a linear optimization is invariant. The final configuration based on this strategy is shown in [Figure 9-18](#). The final configuration agrees well with the published results (See [Figure 9-19](#)) except two small closed loops are formed at the bottom of the structure. Figure 10 is created using Patran's Design Tool/Post-Process/Display Results with the threshold value of 0.2. It should be pointed out that this run does not achieve hard convergence. Instead, it is terminated after 5 consecutive nonlinear analyses unable to find a converged solution.

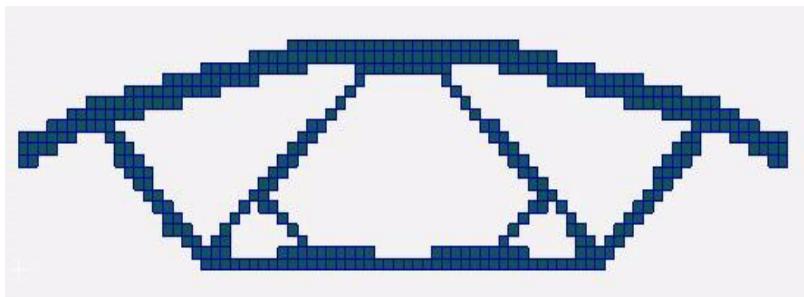


Figure 9-20 Plot of displacement for the alternate updated nonlinear structure

Furthermore, an independent nonlinear analysis is performed for the updated nonlinear FE model under the original loading (deslo10a.dat). The deformation plot from this nonlinear analysis is shown in [Figure 9-21](#) with the maximum displacement of 0.021m that occurs at the loading point.

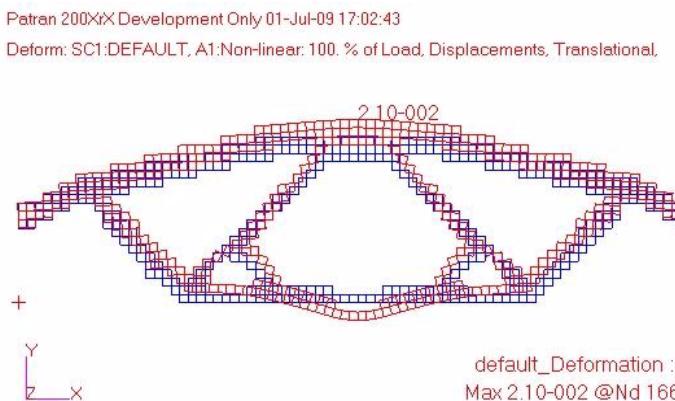


Figure 9-21 Plot of displacement for the alternate updated nonlinear structure

From this example, we have learned that it is important and necessary to perform the topology optimization by including large displacement as nonlinear responses. In addition, if measured in terms of the maximum displacement, both updated nonlinear structures from [Figure 9-17](#) and [Figure 9-20](#) are close, 0.0184m vs. 0.021m. However, their configurations are significantly different. Therefore, the nonlinear topology optimization tasks are more difficult to solve than the linear ones because it not only



tends to show the local minima phenomenon but also the nonlinear analyses may not lead to convergent solution because the collection of elements with lower strain energy or ‘empty’ elements.



Solving Large Problems in SOL 200

This section deals with some specialized techniques that are available in MSC Nastran for solving large finite element tasks. Two techniques, DMP (distributed memory parallel) and ACMS (automated component modal synthesis) are discussed here briefly to show the impact they can make of optimization tasks. Parallel design sensitivity analysis (DSA) can reduce clock time spent in a sensitivity analysis.

Three distributed memory parallel (DMP) techniques are available for SOL 200: internal parallel for FRRD1 module, frequency domain parallel Lanczos method and degrees-of-freedom (DOF) domain parallel Lanczos method.

Internal Parallel for FRRD1 Module

FRRD1 is the solution generator for frequency response analysis. For each forcing frequency, FRRD1 goes through identical steps to produce solution vector(s). Therefore, the work assigned to each processor is equal to the total number of forcing frequencies (NF) divided by the number of processors (NP). If the results of NF divided by NP is not an integer, one or more processors will need to pick up one extra forcing frequency. Hence, the maximum load unbalance among all processors in “internal parallel for FRRD1 module” is the solution time for one forcing frequency. Although this DMP technique is supported in both direct frequency (DFREQ) and modal frequency (MFREQ), it has the biggest impact for DFREQ jobs.

To request NP number of processors for frequency response SOL 200 job, add ‘dmp=NP’ to the job submittal line.

Note: It is possible to select nodes for the DMP job. The defaults for the hosts are machine dependent. See “hosts” in Keywords and Environment Variables (App. C) of your latest MSC Nastran Installation and Operations Guide for more details.

As an illustrative example, a vehicle model with a half million DOFs in the analysis set has been exercised using a varying number of processors. The direct frequency analysis has 12 forcing frequencies. A performance study on an IBM SP3 produced the following results.

Number of Processors	Total Elapsed Time	Speed-up	Elapsed Time in FRRD1	Overhead for DMP in FRRD1
1	252 min. 9 sec.	1	208 min. 4 sec.	0
2	139 min. 48 sec.	1.8	105 min. 44 sec.	9 min. 25 sec.
4	101 min. 29 sec.	2.5	67 min. 16 sec.	16 min. 6 sec.

Overhead for DMP in FRRD1 occurs at the beginning and the end of FRRD1. Note that the overhead can be greatly influenced by the workload of the system and network. The overhead is shown here to indicate that the overhead can become significant and the effectiveness of DMP reduces as the number of processors increases. In this case, the FRRD1 module dominates the overall computing requirements so that there is a significant payoff from running this module in the parallel mode.



Frequency Domain Parallel Lanczos Method

Frequency domain parallel Lanczos in SOL 200 is available for ANALYSIS= MODES and other analyses that use the modal formulation (ANALYSIS=MFREQ, MTRANS, MCEIG and FLUTTER) .

The following Executive Control Statement

```
DOMAINSOLVER MODES (PARTOPT=FREQ)
```

must be added to the beginning of the Executive Control Section.

For example:

```
NASTRAN bufsize=16385 $  
domainsolver modes (partopt=freq) $  
SOL 200
```

Furthermore, the number of processors (NP) must be specified using the 'dmp=NP' keyword on the submittal line. The Lanczos method must be used.

A vehicle model with a half million DOFs in the analysis set is used to indicate of benefits of frequency domain parallel Lanczos. This is an MFREQ job with 12 forcing frequencies. A performance study on an IBM SP3 produced the following results:

Number of Processors	Total Elapsed Time	Speed-up	Elapsed Time in READ Module
1	245 min. 27 sec.	1	178 min. 57 sec.
2	217 min. 45 sec.	1.13	133 min. 16 sec.
4	158 min. 43 sec.	1.55	79 min. 35 sec.

In this case, the READ module is less dominant in the overall total job elapsed time, but there is still a significant speedup with more processors.

DOF Domain Parallel Lanczos Method

The DOF domain parallel Lanczos method is an alternative technique for extracting eigenvalues in parallel mode. SOL 200 also supports this option for ANALYSIS = MODES and other analyses that use the modal formulation (MFREQ, MTRANS, MCEIG, and FLUTTER).

The following Executive Control Statement

```
DOMAINSOLVER MODES (PARTOPT=DOF) $
```

must be added to the beginning of the Executive Control Section.

For example:

```
NASTRAN bufsize=16385 $  
domainsolver modes (partopt=dof) $  
SOL 200
```



The number of processors (NP) must be specified using the ‘dmp=NP’ keyword on the submittal line. The Lanczos eigenvalue method must be used.

A vehicle model with a half million DOFs in the analysis set is again used for this example. This is an MFREQ job and has 12 forcing frequencies. A performance study on an IBM SP3 produced the following results:

Number of Processors	Elapsed Time	Speed-up	Elapsed Time in READ Module
1	245 min. 27 sec.	1	178 min. 57 sec.
2	228 min. 42 sec.	1.07	149 min. 40 sec.
4	188 min. 49 sec.	1.30	95 min. 3 sec.

Parallel Sensitivities

Introduction

Design sensitivity calculations may be performed in a distributed parallel environment in SOL 200 of MSC Nastran. It is a coarse parallel implementation that divides the sensitivity task across a number of processors so that each processes a subset of the total number of design variables. Following the sensitivity analysis and before optimization, the separate sensitivity data are appended into a global sensitivity set.

Benefits

This feature is aimed at users who are currently spending significant time in the sensitivity phase of an optimization design task. This typically occurs when the problem size (g-size) is large, there are many (perhaps thousands) of design variables and the adjoint method of sensitivity analysis is either unavailable or still time consuming.

User Interface

Inputs

- Specify the DOMAINSOLVER Executive Control statement with the new DSA keyword. For example:

DOMAINSOLVER DSA

Other DOMAINSOLVER options like ACMS, FREQ, and MODES may also be specified or modified along with DSA.

For example,

DOMAINSOLVER DSA ACMS

By default, DOMAINSOLVER options MODES and FREQ are always on with dmp keyword on the Nastran submittal command.

- Specify the dmp=n keyword on the Nastran submittal command; where n is the number of available processors.



Outputs

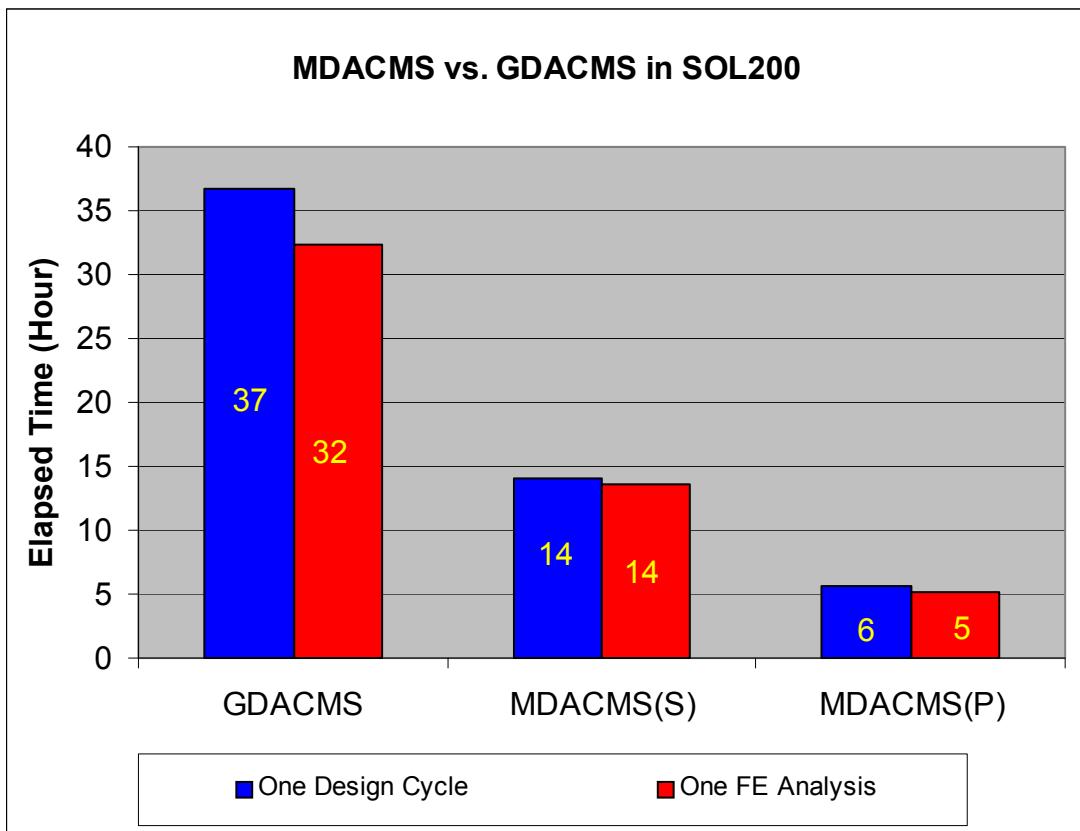
There are no new outputs. It should be noted that with dmp=n, by default, only the master processor's output (f04 and f06) is saved. slaveout=yes may be specified on the Nastran submittal command to request the slaves' output.

Matrix Domain Automated Component Modal Synthesis (MDACMS)

MDACMS is an advanced Lanczos solver introduced in MSC Nastran 2005. You can find a detailed discussion in "ACMS Now Available in the Matrix (DOF) Domain" in the *MSC Nastran 2005 Release Guide*. You can use this feature in a SOL 200 run by specifying the DOMAINsolver ACMS (partopt=dof) command. It is available for Analysis=Modes and/or =MFREQ and both serial and parallel processing are supported. A parallel run is activated when DMP = n and n > 1.

This feature has been tested with an NVH optimization task with a 4.2M dofs, for an eigenanalysis up to 300 Hz that produced 1960 modes. It is not practical to execute this problem without the use of an advanced numerical technique. Three separate runs were performed. The following plot shows that a parallel (4 processor) MDACMS SOL 200 job achieves more than 6 times speedup relative to a Geometric Domain ACMS (GDACMS); DOMAINsolver ACMS (partopt = grid) SOL 200 run (reduced time from 37 hours to 6 hours) while the serial MDACMS SOL 200 run reduces to 14 Hrs for a 2.5 times speedup. GDACMS is now regarded as an obsolete capability.





Special Considerations When Designing One-Dimensional Bending Elements

The seemingly simple BAR and BEAM straight 1-D bending elements in MSC Nastran have some special behaviors that need to be pointed out if they are to be effectively designed. To begin, there are six property types that are associated with these elements:

1. The PBAR defines basic bending properties of a CBAR element.
2. The PBARL defines dimensions of MSC supplied or user-supplied section types for the CBAR element.
3. The PBRSECT enables the definition of arbitrary cross sections for CBAR elements by specification of section type and sets of data that define the boundaries.
4. The PBEAM defines bending properties for a more complete 1-D element in that it allows definition of the shear center, the neutral axis, center of gravity, warping and taper. The PBEAM property entry is referenced by the CBEAM. The taper, in particular, makes the design of the PBEAM somewhat complicated, although MSC has greatly simplified the design model requirements for the PBEAM in recent releases.
5. The PBEAML defines dimension of MSC provided or user-supplied section types for the CBEAM element. The availability of these libraries greatly simplifies the data preparation required from the user. All of the features of the PBEAM, including taper, are available with the PBEAML.
6. The PBMSECT enables the definition of arbitrary cross sections for CBEAM elements by specification of section type and sets of data that define its boundaries.

For all of the discussion of this section, the design is assumed to be performed using the PNAME option for the fifth field of the DVPREL1 and DVPREL2 entries. The alternative specification of an FID or and EPT location is deemed obsolete.

PBAR

The design of the PBAR is straightforward. Typically, the user's design variables are cross-sectional dimensions, necessitating the use of type-2 properties in specifying the properties as a function of the design variables. The following considerations can be mentioned when designing PBARS:

- Only use type-2 properties when they are needed. This is because each type-2 property is treated as an additional design variable in the design sensitivity process (this is discussed in Internal Representation of the Sensitivity Coefficients in MSC Nastran on pp. 53-54). This places a performance and numerical accuracy burden on the sensitivity analysis.
- Keep in mind that changing a beam dimension affects not only the standard properties such as area and moments of inertia, but also the stress recovery locations as well. These latter properties are typically linear in the cross section dimensions and can therefore be designed using the DVPREL1 entry.

The generality of the PBAR makes it possible to make assumptions about the relationships among the section properties. One could postulate, for example, a thin-walled beam where, to a first approximation,



all the properties are linear in terms of the wall thickness. Alternatively, one could select the area of the beam as the design variable and then specify that the bending moment of inertia is proportional to the square root of the area.

PBARL

The design of the PBARL is the easiest from the user's point of view. Typically, the user specifies designed beam dimensions as linear functions of the design variables (nonlinear functions of beam dimensions with respect to the design variables are not permitted) and the rest of the design is performed internally. An important consideration is that these internal calculations can be done in one of two ways. If the MDLPRM bulk data entry has TWBRBML=1, then the beam library formulations are used to evaluate bar properties and their sensitivities. If this parameter is not set, the bar properties are computed using the same finite element technology that is employed in creating these properties for the PBRSECT bulk data entry. There is a tradeoff in whether to use the library approach or the finite element approach in that the finite element method is considered more accurate but the sensitivity calculation, which uses differencing techniques for all the properties, is computationally more expensive. The guideline is that the default finite element approach be used unless performance issues make this prohibitive. The remainder of this discussion deals with issues associated with the beam library approach.

[Adding Your Own Beam Cross-Section Library](#) describes how you can add your own beam cross section types and it provides insight into the operations that must occur to cause a change in the beam dimension to be reflected in the beam properties. Briefly, MSC Nastran determines which properties of the beam cross section are affected by each design variable and constructs what are referred to as "spawned" design properties to be used in the design. These are properties that have not been explicitly designed by the user, but can be inferred from the property/dimension relationship. A determination is made if this property/dimension relationship is linear or nonlinear so as to use the less expensive and more exact linear relationship when it is valid.

A drawback of using the PBARL is that it can produce more designed properties than may be necessary for the design task at hand. For example, the torsional stiffness of the BAR may have no effect on the response of the structure and a user might not include its design when a PBAR is used. The PBARL will always design this property if it is affected by a designed dimension. Perhaps a better example of this is that stress recovery locations that are affected by a designed PBARL dimension are always designed while a user may find it necessary only to define one or two of these points. In most cases, the design of these extra properties is not a large concern, but it does become an issue when hundreds of PBARL's are being designed in a large finite element model.

PBRSECT

Arbitrary Beam Cross-section

Arbitrary beam cross-section, ABCS, capability has been implemented via Bulk Data entries, PBRSECT and PBMSECT. There are three FORMs to describe arbitrary beam cross section with PBMSECT/PBRSECT, namely GS, CP and OP. For design optimization analysis, data items from PBRSECT and PBMSECT entries can be referenced by the design variable property relation entries, DVPREL1/DVPREL2. Data items that can be considered as design variables in the design optimization analysis include:



- Overall Width - input W for PNAME field of DVPREL1/2. Overall width is computed as X1max-X1min. Both X1max and X1min are obtained by examining X1 field of all POINT entries referenced by a PBMSECT/PBRSECT. This data item is available for GS, CP, and OP.
- Overall Height - input H for PNAME field of DVPREL1/2. Overall width is computed as X2max-X2min. Both X2max and X2min are obtained by examining X2 field of all POINT entries referenced by a PBMSECT/PBRSECT. This data item is available for GS, CP, and OP.
- Overall or Segment Thickness - input T or T(id) for PNAME field of DVPREL1/2. This data item is available only for CP and OP.

Following response types can be utilized for design constraints,

- RTYPE=STRESS for regular stress recovery points of BAR and BEAM element, namely C, D, E and F. Note that the coordinates of the stress recovery points are selected by internal logic which pick POINTs with extreme coordinates. Use item code for CBAR(34) and CBEAM(2).
- RTYPE=ABSTRESS for screened BAR/BEAM element stresses. ABSTRESS is computed only for BAR and BEAM elements referencing PBRSECT and PBMSECT, respectively. Use item code for CBAR(238) and CBEAM(239).

Guidelines and limitations:

- New PBRSECT, PBMSECT, and POINT entries can be punched after each design cycle. Note that connecting overall-Width and/or overall-Height to design variables has the effect of changing the location of POINTs defining an ABCS, even though POINT itself can't be linked directly to design variable.
- FEM is utilized to compute cross sectional properties of ABCS. During internal iterations of optimizer, FEM solution process is used repeatedly to compute properties of new designs. The cost (CPU time spent) of optimizer for a job with ABCS as design variables can become prohibitively high. To alleviate the cost concern, 'PARAM,DV3PASS' can be utilized to control the frequency of re-evaluation of sensitivity coefficients of ABCS sectional properties.
- Current implementation of optimization does not support composite ABCS via 'PARAM,ARBMSTYP,TIMOSHEN(or TIMOFORC)'.

PBEAM

The design of the PBEAM is complicated by the possibility that the beam can be tapered. The original requirements for designing the PBEAM were quite involved and non-intuitive, but MSC has simplified these requirements so that the design of constant section PBEAMs follows the same basic procedures as those for the PBAR. Older documentation that refers to the need to, for example, design both ends of a constant section beam is now obsolete and is superseded by this document.

When designing a tapered beam property, the station is identified in parenthesis. As examples, specifying A(A) or A on a DVPREL1 or DVPREL2 entry indicates that area of end A of the beam is to be designed, I1(B) indicates that the plane-1 inertia at end B is to be designed while D2(6) indicates that the z-location at the D stress recovery point for the 6th station is being designed.

As mentioned, the design of constant section beams is now quite similar to that for the PBAR. It should be kept in mind that the user input for the PBEAM is used to determine whether a beam can be tapered.



If only END A data are provided on the PBEAM entry, the beam is regarded as constant section and only END A properties can be designed. Similarly, if END B or other station data are provided on the PBEAM entry, the beam is regarded as tapered and care must be taken to explicitly design each property that is expected to vary. As a final comment, if the PBEAM contains specification of SO and X/XB for END B and other stations but blank fields are used to specify the properties, the beam is still regarded as a constant section. This input can be used when it is desired to obtain stress output for a constant section beam at an intermediate station.

PBEAML

The design of the PBEAML is much simpler than that of the PBEAM in that it is only necessary to indicate which beam dimensions are to be changed and the beam library performs the remainder of the design task. [Adding Your Own Beam Cross-Section Library](#) can provide insight into how this design occurs. As with the PBARL, it should be kept in mind that all the properties that are affected by the designed dimension will be designed as well, regardless for whether the user feels this is important. For example, the DIM3 dimension that specifies the height of an “H” type element affects the C1,D1,E1 and F1 stress data recovery locations in a linear way and the A,I1,I2,J,K1,K2,CWA and CWB in a nonlinear fashion. It is likely that some of these properties are not important in the design or perhaps the stress recovery points are redundant, but they will be designed in any case. Again, this is not a significant issue unless the models are large with many designed PBEAML entries.

The PBEAML can be tapered and it is the PBEAML entry that governs whether the beam is permitted to taper. Like the PBEAM entry, if the PBEAML provides data at END B and/or other intermediate stations, it is considered tapered. If there is only END A data, the beam is considered constant section and only END A dimensions can be specified on DVPRELi entries. If the PBEAML is tapered, only dimensions explicitly invoked on DVPRELi entries will be designed. One distinction between the PBEAM and PBEAML is with regard to taper. As mentioned above, the PBEAM is considered a constant section if the only inputs at END B and/or intermediate stations are SO and X/XB. For the PBEAML, if any input (including SO and X/XB) is provided at END B or an intermediate station, the beam is considered tapered.

PBMSECT

The description of the PBRSECT above applies to the PBMSECT as well and are not repeated here. Unlike the PBEAML, the PBMSECT does not support taper. The PBMSECT supports using composite materials in its construction, but these features are not supported for design.



MultiOpt-Global Optimization and Multi Model Optimization

Introduction

The MSC Nastran utility MultiOpt was released in MSC Nastran 2010 to support Multiple Model Optimization (MMO) that combines two or more related optimization tasks into a single combined optimization task. In the MSC Nastran 2016 release, a new feature, Global Optimization (GO), was added to the MultiOpt program. In addition, MMO was enhanced to support Mode i8 computer platforms and remove the limitation of up to five MSC Nastran models.

Global Optimization

MSC Nastran has been supporting structural optimization via SOL200 for many years. The technology built in SOL 200 uses gradient based local optimization algorithms. Many industry optimization applications show that multiple local minima are often obtained when starting from differing initial designs. This behavior is particularly evident in composite and dynamical response optimizations. Clients often want to know whether the optimal design is a local or global optimum. If it is a local, how can they improve the local optimal design with limited computer resources? Mathematically speaking, it is impossible to find a global optimum for general nonlinear problems unless all the possibilities in the design space are exhausted. However, numerically, it is desirable to have a practical global optimization procedure to find an approximate global solution with reasonable computing cost. MSC Software has developed a multi-start method to solve both unconstrained and constrained global optimization problems. It has four major aspects:

- The global design space is first approximated by a small but intelligent set of points that are derived using Design of Experiment (DOE) methods, such as orthogonal arrays, fractional or full factorial design techniques. This important step enables the subsequent local searches to be performed in a well-balanced, but reduced, design space.
- Heuristic techniques are applied to quickly find the global optimum. The basic rule is to always start a new local search from the point that has the largest distance from the current best local minimum point. A direct benefit of this is to minimize the number of repeated visits to the same region of attraction.
- A gradient-based optimization tool is used as the local search engine. Specifically, the powerful multidisciplinary design optimization capability in MSC Nastran SOL 200 is used here
- Asymptotically correct condition or a user specified time budget is used to terminate the global search process.

More discussion of the method can be found in a paper titled "A Practical Global Optimization Procedure", AIAA-2003-1671, 2003.

Multi Model Optimization

The MMO application starts the processing of the separate design models up to the point where the optimization is to occur. Then, it merges the design information, performs the optimization and partitions



the results. Then the individual models are resumed in a design loop that is terminated when either convergence is achieved or the maximum design cycles are reached. A flow chart of this process with two design models is given in [Figure 9-22](#).

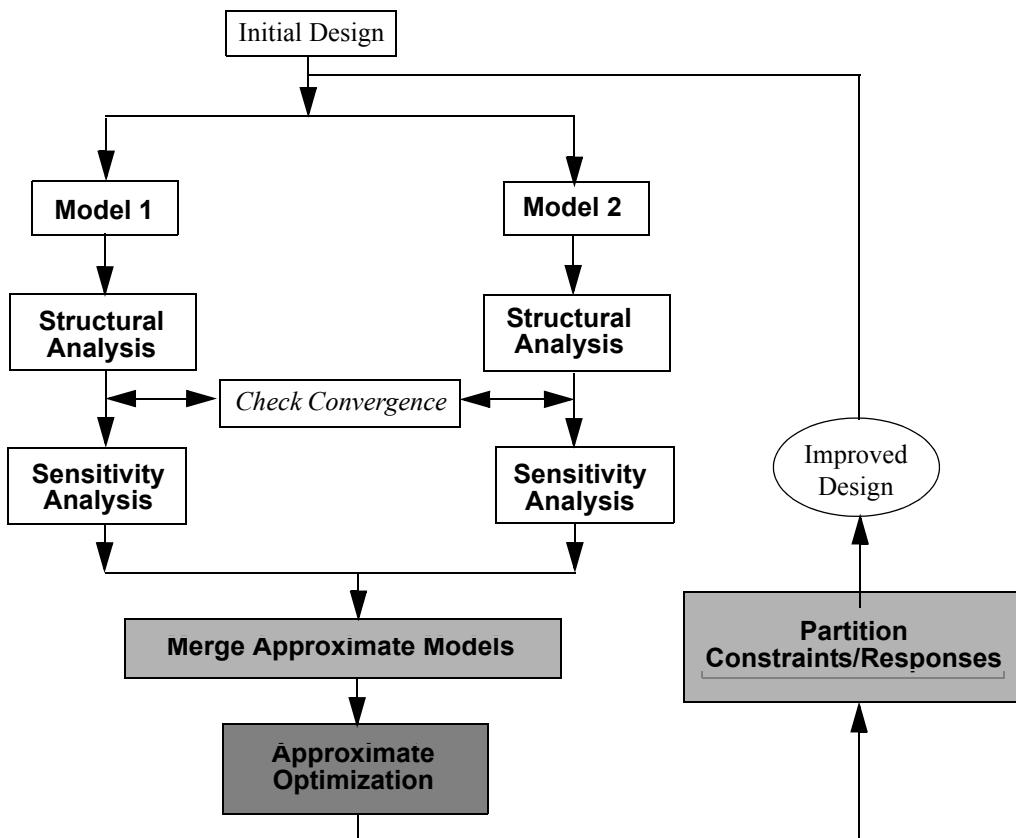


Figure 9-22

The boxes ‘Merge Approximate Models’, ‘Approximate Optimization’, and ‘Partition Constraints/Responses’ are driven by a subDMAP that is invoked by a MultiOpt created input deck ‘**MERGE.dat**’ (i.e, the user does not need to create or even know about this deck).

Benefits

The benefit to the user for the automated global optimization (GO) feature is that it combines automatic multi-start global methods and gradient based local optimization algorithms. The heuristics built into the global search algorithm allow users to explore better designs with relatively modest computing resources. It is expected that certain applications of applying GO will provide surprising variability in the design task and this would provide deeper insight design process. GO can also be utilized to search for a feasible design when a single SOL200 optimization was unable to find one.



The MMO ability to merge multiple design models allows the user to have separate models that differ in their topology or in their analyses but still perform a combined optimization.

One scenario might be if different variants of an aircraft share a common wing design but have different fuselages (standard and stretch configurations). It would be difficult to design this with the standard version of SOL 200, but it is readily achievable with this capability.

Another scenario is one where a detailed model of a component is available and can be used to match analysis data with test results. At the same time, a less detailed model of the same component is included in a global model that is being used to perform optimization with global constraints, such as flutter speed.

MMO can be used in this case to perform both optimizations simultaneously and thereby maintain a common design.

A final scenario addresses the common scenario in the design stage where different groups have different finite element representations of the same vehicle or device. For example, the stress group might have a detailed model suitable for detailed analysis while the vibration group may have a less detailed model to be used in providing modal information. With MMO, it is possible to retain these two separate models but still apply design optimization techniques. For this to be of value, there needs to be some common design variables that are shared by all the models and there needs to be a single objective that is based on responses from one or more of the models. The constrained responses can be applied completely separate from each other, in the separate models.

User Input

For GO applications, a standard MSC Nastran SOL 200 job can be used to explore global optimization.

For MMO applications, each of the models has its own data deck that is complete to the extent that it could be run "stand-alone" in MSC Nastran SOL 200. The following changes need to be made in each of these separate input files for a MultiOpt submittal:

1. A DESMOD case control command needs to be inserted in each model of the form:

```
DESMOD = model
```

2. Where "model" is a unique, user specified name of up to eight characters that designates the model. The DOPTPRM entry in the separate models control parameters that are used by the optimizer and others that are used in the iterative process. It is recommended that an identical DOPTPRM be used in each model. The DOPTPRM in the first model controls what is used by the optimizer; e.g., OPTCOD, while convergence parameters and DESMAX are controlled by the local DOPTPRM entry.
3. If the objectives are to be combined, the first model must have its DESOBJ request point to a DRESP2 that contains the structure of the combined objective function. This is expanded upon below.

Invoking MultiOpt

GO and MMO are features in the program MultiOpt. The way to invoke MultiOpt is using a command line of the following form:



MSC20170 MultiOpt mygofile.xml (or mymmofile.xml)

User GO Input

The GO input xml file (mygofile.xml) has this format

```
<?xml version="1.0" ?>
<rc OptType="GO" debug="no">
<Job name="deck" MEM="m" SMP="n" SCR="s" blocking="bopt"/>
<goparam variability="gopt" maxfea="mopt" nglsmx="nopt"
fsave="fopt" minmax="mmopt"/>
</rc>
```

For example,

```
<?xml version="1.0" ?>
<rc OptType="GO" debug="no">
<Job name="pcomp" mem="2GB" SCR="yes" blocking="3"/>
<goparam variability="uplo" maxfea="100" nglsmx="5"
fsave="1"/>
</rc>
```

where:

Option	Description
OptType	OptType="GO" must be used to perform global optimization
Debug	"No" or "yes" (default = no). If turn on debug="yes", the program will produce more message for diagnosis
Job name="deck"	Indicates the subsequent data is a job name. deck is the basename of the input data model (deck.dat or deck.bdf). No default
MEM= "m"	Indicates the subsequent data is a memory request for this model. Integer m = amount of memory requested (default=1GB)
SMP="n"	Indicates the number of processors used to efficiently obtain a solution to the simulation. The processors used will be on a single machine (default = no use of parallelization on shared memory tasks).
Scr="s"	Option for scratch. Character s = yes (default) or no. Scr="yes" is recommended.
Blocking="bopt"	Indicates whether jobs run in parallel or serially. Integer bopt =1(default) runs the job serially while bopt=n enables running n local optimization jobs in parallel.
goparam	Indicates the job has user specified GO control parameters (optional)



Option	Description
Variability	<p>Specifies how the global design space is sampled.</p> <p>=Golden (default) the global design space is sampled at the points determined by Golden Section Ratios. This parameter is only applicable when the number of design variables is greater than 2. When NDV <=2, the design space is always sampled at the lower and upper limits of each design variable.</p> <p>=Uplo the global design space is evaluated at the lower and upper limits of each design variable.</p> <p>(Note: user must define proper limits on the DESVAR)</p>
Maxfea	Indicates the maximum allowable number of finite element analyses. Integer >0 (default=500). This parameter controls the global search process. The procedure computes the total number of finite element analyses that is the sum of the number of finite element analyses performed for each local search. When the total number of FE analyses exceeds mopt, the GO process is terminated with the appropriate message
Nglsmax	Indicates the maximum allowable number of consecutive local searches that do not produce an improved design. Integer (default=8). This parameter also controls the global search process. The global search procedure will be terminated when the number of consecutive local searches that do not produce an improved design exceeds the given nopt parameter.
Fsave	By default (fsave="0"), only the initial local optimization and the global optimization results are saved. If the user wants to save intermediate local optimization results that have feasible designs, fsave can be set to fsave="1"
Minmax	By default (minmax="0"), GO explores global minimum objective. Otherwise, minmax="1" searches for a global maximum objective.

User MMO Input

The user MMO input xml file (mymmofile.xml) has this format

```

<?xml version="1.0" ?>
<rc OptTYpe="MMO" debug="yes">
  <Job name="deck1" coef="c1" MEM="m1" SMP="n1" SCR="s1"
blocking="b1"/>
  <Job name="deck2" coef="c2" MEM="m2" SMP="n2" SCR="s2"
blocking="b2"/>
  .
  .
  .
  <Job name="deckn" coef="cn" MEM="mn" SMP="nn" SCR="sn"
blocking="bn"/>
  <Merge MEM="mm" SMP="nm" SCR="sm" />
</rc>
```



For example,

```
<?xml version="1.0" ?>
<rc OptType="MMO" debug="yes">
  <Job name="pcompl" coef="1.0" MEM=2GB" SMP="2" SCR="yes"
  blocking="1"/>
  <Job name="pcomp2" coef="1.0" MEM=2GB" SMP="2" SCR="yes"
  blocking="1"/>
  .
  .
  <Job name="pcompn" coef="1.0" MEM="2GB" SMP="2" SCR="yes"
  blocking="1"/>
  <Merge  MEM="1GB" SMP="2" SCR="yes" />
</rc>
```

where:

Option	Description
OptType	OptType = "MMO" is used to perform multiple model optimization
Debug	"No" or "yes" (default = no). If you turn on debug="yes", the program will produce more message for diagnosis
Job name="deck1" (deck2,...,deckn)	Indicates the subsequent data is a job name. deck 1 is the basename of the input data model. deck1, deck2, ..., and deckn are n basename in a multiple mode optimization problem
coef= "c1" (c2,...cn)	Indicates the subsequent data is a coefficient for this model. c1 is the coefficient that provides the objective weighting for the first model (default=1.0). c2, ...c2n are coefficients that provide the objective weighting for the rest models (default=0.0)
MEM= "m1" (m2,..mn, mm)	Indicates the subsequent data is a memory request for this model. Memory for the model m1, m2, ..., mn, mm = 1GB (default). mm is the memory for the merge operation
SMP="n1" (n2, ...nn, nm)	Indicates the number of processors used to efficiently obtain a solution to the simulation (default = no use of parallelization on shared memory tasks).
Scr="s1" (s2, ..., sn, sm)	Option for scratch. scr=yes (default) or no. Scr=yes is recommended.
Blocking="b1"(b2, b3, .., bn)	Indicates whether that job is to be run in parallel or serially. The default runs the job in parallel. To run MMO sequentially, set blocking="0" for each MMO jobs.
Merge	Indicates a solution for the merge job (optional)

Remarks

1. MultiOpt supports Mode i8 and new IFP only.
2. MultiOpt input XML file (for example, mygofile.xml and mymmofile.xml) is a standard XML file. Users can use any XML editor and validation tools to create, validate, and correct.
3. OptType = "GO" must be used to perform global optimization. The goparam (global optimization control parameters) are optional.



4. GO search is terminated when any of three conditions is met:
 - a. all the design points in the global search space are exhausted.
 - b. The number of consecutive local searches that do not provide an improved design exceeds the given NGLSMX number.
 - c. The total number of FE analyses exceeds the given MAXFEA number.
5. OptType = "MMO" is used to perform multiple model optimization. The merge job in the MultiOpt input XML file is optional.
6. MMO is terminated when either convergence is achieved or the maximum design cycles are reached for each separate models.
7. More discussions about the options MEM, SCR and SMP can be found in MSC Nastran Quick Reference Guide.
8. MMO combined objective function

COEF = Coefficient that provides the objective weighting for the model (default: =1.0 for the first model and 0.0 for subsequent models). This default means the objective comes from the first model only. When you are optimizing across several models, the overall objective can simply be the objective of the first model or it can be a linear combination of the objectives from all the models. If the objective is only from the first model, you set up the objective in this model in the same way as a standalone SOL 200 optimization task and not apply any coefficients (coef and ci data in the above options) in the xml file. Combining objectives from several models relies on a DRESP2 in the first model that has to be constructed with some specific rules:

- The DRESP2 must have a "DTABLE" with nmod (the number of models) LABL1 values. The value specified for the first LABL1 should be 1.0 and the subsequent LABL1 values should be zero. In this way, the objective of the first model is simply the first response. (Other LABL1 values in the first model are permitted, but can lead to confusion)
 - The DRESP2 must have either nmod DRESP1 quantities or nmod DRESP2 quantities. It is not possible to mix DRESP2 and DRESP1 responses in the combined objective.
 - The second through nmod responses in the combined objective of the first model are dummy and will be replaced with the second through nmod responses that are the objectives in the second and subsequent models.
 - The coefficients actually used for the models are the Ci data provided in the xml file.
 - The DEQATN that corresponds to the DRESP2 performs a linear addition of the responses:
 - COBJ = Ci OBJi
9. MultiOpt users can also use the Nastran RC file to set other Nastran keywords except (MEM, SMP, and SCR) for GO or MMO. See the MSC Nastran Quick Reference Guide.
 10. Windows users

In order to run the MultiOpt utility, remotebootstrap.exe should be added to Windows Firewall access list. Otherwise, it will be blocked by it. When MultiOpt is run, users should not disturb the command prompt, especially when there are a large number of jobs;, i.e., more than 5. The status of execution should be monitored from another prompt. Otherwise, the original one may be unresponsive.



11. The folder path cannot contain blank spaces.

User Output

There are four types of outputs from GO.

- A single summary output file multiopt.log that records the history of the search process. The local optimization and final global optimization results can be checked in that file.
- fn_go.dat is the global optimization job (fn is the basename of input model) with a sample design that produces the global optimal solution. The sample design (as a design variable include file) fn_xxxx_desvar.go is also saved for the global optimization.
- fsave="1" can be used to keep all local optimization results with a feasible design. fn_xxxx_desvar.go is a design variable include file containing DESVAR entries where xxxx is a four digit number indicating a particular design run. fn_xxxx.f06 is the local optimization result for the run.
- Patran (or third part post-processor) can be used to view the global optimal design and optimization history from the best local optimization run.

For MMO, the separate models each provide their separate .f06 files and any other outputs that would be expected from a SOL 200 run. There is also a merge.f06 file produced from the optimization portion of the process and could provide some insight into how the optimization is progressing and highlight any problems.

Comma Separated Values File

The ASSIGN statement (in File Management) can be used to generate optimization history in Comma Separated Values (CSV) file (see [Comma Separated Values File, 303](#)). This is still valid for GO (local optimization) and MMO (separate models). To produce the merge optimization design history in CSV file, users must define USERFILE name by the DESMOD model name in Case Control (for example, DESMOD = model)

```
ASSIGN USERFILE='model.CSV', STATUS=NEW, FORM=FORMATTED, UNIT=52
(It is necessary to include a PARAM, XYUNIT, 52 in the Bulk Data file)
```

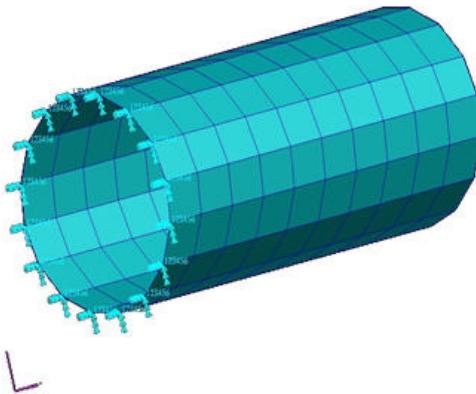
The merge optimization history CSV file is called mymmofile_mmo.csv where mymmofile is the basename of the MultiOpt input XML file.

GO Test Case

In order to validate the GO capability, a problem is obtained from the test problem library for Sol 200 (tpl/optim/d200c01.dat). The following figure shows a composite tube clamped at one end and free at the other end. The tube is built from a composite material with 8 plies. The outer layer pairs are initially at 85° orientations to the hoop direction while 4 core layers are at 60° orientations. The design task is to find the ply thickness and ply angles to minimize weight subject to Hill Failure criteria. Notice that the outer layer pairs and four core layers are two design variables while the ply thickness is the third design



variable. Originally, when the problem was solved with SOL 200, a local minimum solution was found. After the global optimization procedure is exercised, a global optimum solution has been found. See the following Result section for the detailed numbers.



The MultiOpt input XML file can be found in the test problem library (tpl/go_opt/ d200c01.xml) as:

```
<?xml version="1.0" ?>
<rc OptType="GO" debug="no">
  <Job name="d200c01" blocking="2"/>
</rc>
```

The summary file multiopt.log from this example problem is described below to provide a general discussion of the various output section in the file. Since the other types of files are basically regular Nastran output file, they are not discussed here.

The section of Initial local search

This section shows the result from the very first local search that is based on the original user input file. First, the header indicates initial local search. Then, it tabulates three design variables with the starting and final values, bounds and labels. Finally, the objective and maximum constraint values are shown. This final design is marked as a feasible solution because the constraint value is less than the gmax (a parameter defined on DOPTPRM entry and its default is 0.005). The number of FE analyses used in this local search is also given.



```
*****
*** Initial Local Search ***
*****
```

The starting design variables are from your input deck

desvar ID	Label	Starting Value	Final Value	Lower Bound	Upper Bound
1	TPLY	1.0000E+00	5.7385E-02	1.0000E-03	1.0000E+01
2	THETA	1.0000E+00	5.6849E-01	-1.0588E+00	1.0588E+00
3	THETA	1.0000E+00	1.6880E-01	-1.5000E+00	1.5000E+00

THIS IS THE FIRST LOCAL SEARCH BASED ON THE USER INPUT FILE
 OBJECTIVE = 9.1709E-02,
 MAXIMUM CONSTRAINT VALUE = 3.9210E-03 (A FEASIBLE SOLUTION).
 THE LOCAL SEARCH USED 17 FE ANALYSES.

Note that the actual ply angles are plus or minus 85.0 times the second design variable and plus or minus 60 times the second so that the plies can range from minus 90 to plus 90 degrees. Allowing ply angles to vary over such a broad range is known to result in local minima.

The section of subsequent local searches that produce an infeasible solution

This section shows the result from the local search #1. It is actually the second local search if the very initial local search is included. The statement following the header indicates that design run # 7 from the design table is used in this local search. If a local search produces an infeasible solution, the output section will indicate it accordingly, as shown here. Basically, the global search procedure simply discards such points and proceeds to another local search based on the new design run.

```
*****
*** Local Search # 1 ***
*****
```

The starting design variables are from sample # 7

desvar ID	Label	Starting Value	Final Value	Lower Bound	Upper Bound
1	TPLY	7.5003E+00	2.3414E-02	1.0000E-03	1.0000E+01
2	THETA	-5.2941E-01	-3.6165E-02	-1.0588E+00	1.0588E+00
3	THETA	-7.5000E-01	-6.9144E-02	-1.5000E+00	1.5000E+00

THIS LOCAL SEARCH DOES NOT PRODUCE A UNIQUE LOCAL MINIMUM.
 OBJECTIVE = 3.7419E-02,
 MAXIMUM CONSTRAINT VALUE = 8.9975E-03 (AN INFEASIBLE SOLUTION).
 THE LOCAL SEARCH USED 21 FE ANALYSES.

The section of subsequent local searches that produce a feasible solution

This section shows the result from the local search #5. The statement following the header indicates that design run # 9 from the design table is used in this local search. The starting and final values of three design variables from search # 5 are listed under the columns of starting and final value. See the highlighted sentence A FEASIBLE SOLUTION. It is possible that a local search may be trapped in the same region of attraction and the final design will be identical to a previous local minimum. When that happens, the wording in the statement is different.



```
*****
          Local Search #      5
*****
The starting design variables are from sample #      9

```

desvar	ID	Label	Starting Value	Final Value	Lower Bound	Upper Bound
	1	TPLY	5.0005E+00	2.1409E-02	1.0000E-03	1.0000E+01
	2	THETA	0.0000E+00	-6.2499E-03	-1.0588E+00	1.0588E+00
	3	THETA	0.0000E+00	-6.2499E-03	-1.5000E+00	1.5000E+00

THIS LOCAL SEARCH PRODUCE A CURRENT BEST LOCAL MINIMUM
OBJECTIVE = 3.4214E-02,
MAXIMUM CONSTRAINT VALUE = 2.0041E-03 (A FEASIBLE SOLUTION).
THE LOCAL SEARCH USED 16 FE ANALYSES.

The final output section

The final section shows the global solution with the best objective and the maximum constraint value. The user information message explains how the global search procedure is terminated. In this example case, the job is terminated when all the design points in the design table are invoked. It also shows the total number of FE analyses and the number of local searches being used in this job. The history of local minima shown at the end of the section can be used for an xy-plot to gain the design insight (via *.xdb)

```
0*** USER INFORMATION MESSAGE
RUN TERMINATED DUE TO ALL THE DESIGN POINTS IN THE SAMPLE SPACE ARE INVOKED.
THIS RUN USED    201 FE ANALYSES AND      9 LOCAL SEARCHES.
THE GLOBAL SOLUTION IS: LOCAL OPTIMIZATION SAMPLE #      9
OBJECTIVE        = 3.4214E-02,
MAXIMUM CONSTRAINT VALUE = 2.0041E-03 (A FEASIBLE SOLUTION).
```

```
***** HISTORY OF LOCAL MINIMA *****
```

RUN	#	LOCAL MINIMUM
	0	9.1709E-02
	9*	3.4214E-02

```
*****  

END OF GLOBAL OPTIMIZATION SEARCH  

*****
```

MultiOpt is successfully complete

```
##### [END MULTIOPT: 03/04/2015 10:53:16] #####
```

MMO Test Case

This section provides details on an example that is both simple and exercises many of the developed features. The example is based on the tpl deck dsoug1.dat and is the classic three bar truss example. The MultiOpt version is located in the tpl/multiopt directory. The dsoug1.dat example has two subcases. For purposes of this demonstration, the deck has been divided into two separate models, each with a single subcase. The first deck is named cobj and contains the design model as it is in dsoug1 and applies only the second subcase. Also, the DLINK entry has been removed (this was for the purposes of the test and not a requirement). The second deck is called sub2d and applies the first load case. In another difference from the dsoug1.dat deck, the cobj deck has a DESOBJ that points to a DRESP2 that is to be used to



combine the two weight objectives from the separate models. (This is a contrived objective that demonstrates how to combine objectives but does not have any physical meaning). The following listing shows portions of the input stream for cobj.dat with the items of interest to this capability highlighted in red.

```

ID MSC, D200X1 $ V68 G_MOORE 24-MAR-1994
TIME 10 $
diag 8,15 $
SOL 200 $ OPTIMIZATION
CEND
TITLE = SYMMETRIC THREE BAR TRUSS DESIGN OPTIMIZATION - D200X1
SUBTITLE = BASELINE - 2 CROSS SECTIONAL AREAS AS DESIGN VARIABLES
label = this run only contains the second subcase of test deck d200x1
$ it is run in conjunction with sub2d.dat
ECHO = SORT
SPC = 100
DISP(plot) = ALL
STRESS(plot) = ALL
DESOBJ(MIN) = 200 $ (DESIGN OBJECTIVE = DRESP2 ID)
DESMOD=SUB1
DESSUB = 21 $ DEFINE CONSTRAINT SET FOR BOTH SUBCASES
ANALYSIS = STATICS
SUBCASE 2
LABEL = LOAD CONDITION 2
LOAD = 310
COBJ = ?Ci ? OBJ
Chapter 9: Special Topics 665
Multi Model Optimization (MultiOpt)
BEGIN BULK
$...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
$ $...DEFINITION OF DESIGN VARIABLE TO ANALYSIS MODEL PARAMETER RELATIONS
$DVPREL1, ID, TYPE, PID, FID, PMIN, PMAX, C0, , +
$+, DV1D1, COEF1, DV1D2, COEF2, ...
DVPREL1 10 PROD 11 4 0.0 +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
dresp2 200 cobj 200
dtable coef1 cdum2
dresp1 20 21
dtable coef1 1.0 cdum2 0.0
deqatn 200 cobj(c1,cdum2,obj1,odum2) = c1 * obj1 + cdum2 * odum2
$ ENDDATA

```

The DESVAR/DVPREL1 Bulk Data entries set up design variables for each of the three rod elements in this case. The DESMOD=SUB1 identifies design model as SUB1. The DESOBJ = 200 points to the DRESP2 that is shown at the bottom of the extracted data. It is seen that the objective is perceived as being the weighted sum of a DRESP1 that is the weight response plus a DRESP1 that is a displacement response. Since the weighting terms are 1.0 and 0.0, the objective for this model is actually the weight of



the model, but this sets up the combined objective for the merged model. The second model is input from the sub2d.dat input and has a different loading condition and a different (i.e., DESMOD=SUB2) DESMOD value.

The MultiOpt input XML file for this job is of the form:

```
<?xml version="1.0" ?>
<rc OptTYPe="MMO" debug="no" >
  <Job name="cobj" coef = "1.0" mem="200m" scr="yes" />
  <Job name="sub2d" coef = "0.5 " mem="200m" scr="yes" />
</rc>
```

The coefficient 1.0 0.5 replaces the DTABLE terms of the cobj.dat shown above so that the combined objective is all of the weight of the first model plus half the weight of the second. Again, this is a contrived objective with no particular physical meaning.

Performing this task results in three .f06 files: cobj.f06, sub2d.f06 and merge.f06. The first two contain the results from exercising the individual models and each has a final design optimization history. The final design achieved by this process is identical to the one shown in [Listing 8-2, Selected Output for DSOUG1](#) (p. 495) in the *Design Sensitivity and Optimization User's Guide*.

Guidelines and Limitations

It is recommended that the associated SOL jobs (for MMO, without DESMOD case control entry) be run to insure that the models are well posed before launching GO/MMO applications. In addition, here are some guidelines/limitations.

GO Guidelines

1. It is recommended to use all default values for the first try (SOL200 DOPTPRM default values except DESMAX, plus GO parameter default values).
2. SCR=yes is recommended since less disk space is required
3. Parallel runs and/or SMP>=2 require more memory. GO may fail due to insufficient memory.
4. The upper and lower bounds on design variables (DESVAR) must be defined properly since they are used to generate sample points. Otherwise, local optimization SOL200 job may fail due to improper sample points.
5. To explore more design space, users can use larger NGLSMAX and/or MAXFEA. In addition, users can have both variability="Golden" and "UPLO" run, or modify design variable limits.
6. Shape optimization is a challenge for GO since there is a higher chance to have bad mesh due to a large shape changes during a GO search. A good quality of shape vectors and proper upper and lower bounds on design variables are essential for global shape optimization. In addition, (DVGRID) must be defined in the basic coordinate system.
7. GO does not support TOMVAR, TOPOPT, BEADVAR and Part Super Element.
8. No design variables are allowed in any include files.
9. Does not support old IFP, supports mode i8 only.



10. The folder path cannot contain blank spaces.
 11. To explore more design space, users can use larger NGLSMAX and/or MAXFEA. In addition, users can have both variability="Golden" and "UPLO" run, or modify design variable limits.
 12. It is recommended that you remove old result files in your working directory when starting a new run.
 13. Optimization, including Global Optimization is not available with FEA licensing.
 14. To support PEM(Poro-Elastic Material), users need to define additional environment variables.
- ```
export MSC_ISHELLPATH=install_dir/msc20170/actran/linux64/Actran_16.1.b.92885/bin
export ACTRAN_PATH= install_dir/msc20170/actran/linux64
export ACTRAN_PRODUCTLINE= install_dir /msc20170/actran/linux64/Actran_16.1.b.92885
export ACTRAN_AFFINITY=reset
export ACTRAN_MPI= install_dir
/msc20170/actran/linux64/Actran_16.1.b.92885/mpi/intelmpi
```

On Windows change "export" to "set" and "linux64" to "win64".

## MMO Guidelines

1. Each model must have a unique DESMOD case control entry.
  2. Design variables that are to be shared across models must have the same ID.
  3. It is an error to have Design Variables in the separate models that share an ID but have different label, XLB,XUB , XVAL or DDVAL information.
  4. Designed properties are assumed to be unique even when they are not. This is because it is almost impossible to verify that the design properties are actually identical (e.g., a DVPREL1 that has identical input may actually be referring to a completely different PSHELL entry in terms of its physical meaning).
  5. Support for shape optimization is minimal in that if the separate models must have the same topology.
  6. The feature that allows the user to define an overall objective that is the weighted sum of individual objective imposes the additional requirement of defining an objective in the first model that represents the desired final model, but with dummy coefficients and objectives. This is described in the input section above.
  7. MMO does not support TOMVAR, TOPOPT, BEADVAR, TREGION=1 and Part Super Element.
  8. It is recommended that you remove old result files in your working directory when starting a new run.
  9. The folder path cannot contain blank spaces.
  10. Optimization, including Multi-model Optimization is not available with FEA licensing.
  11. To support PEM(Poro-Elastic Material), users need to define additional environment variables.
- ```
export MSC_ISHELLPATH=install_dir/msc20170/actran/linux64/Actran_16.1.b.92885/bin
```



```
export ACTRAN_PATH= install_dir /msc20170/actran/linux64  
export ACTRAN_PRODUCTLINE= install_dir /msc20170/actran/linux64/Actran_16.1.b.92885  
export ACTRAN_AFFINITY=reset  
export ACTRAN_MPI= install_dir  
/msc20170/actran/linux64/Actran_16.1.b.92885/mpi/intelmpi
```

On Windows change “export” to “set” and “linux64” to “win64”.

Documentation Dependencies

More information on the DESMOD Case Control command can be found in [DESMOD](#) (p. 260) in the *MSC Nastran Quick Reference Guide*.



OpenMDO™

In MSC Nastran, there are two MSC proprietary general purpose optimizers: MSCADS and IPOPT. In the MSC Nastran 2011 release MSC created a Simulation Component Architecture (SCA) plug-in to allow users access to other commercially available optimization codes or their own optimization codes. The thinking was that some investigators may conjecture that their algorithm for performing optimization surpasses the proprietary algorithms and this provides an opportunity to test and perhaps commercialize their offering in conjunction with SOL 200. Alternatively, some researchers may simply be interested in how their algorithm performs in a commercial environment compared to the proprietary code across the wide variety test cases that MSC has available.

Since no one has taken on the challenge of plugging in an alternative optimizer, it is felt unnecessary to add the description on how to achieve this here. Instead, you are referred to the extensive documentation on OpenMDO™ in the MD Nastran 2011 and MSC Nastran 2011 Release Guide.





A

Nomenclature, Glossary of Terms, and References

- Nomenclature 682
- Glossary of Terms 685
- References 693



Nomenclature

.	Vector dot product
\times	Vector cross product
*	Scalar multiplication
$ a $	Absolute value of the quantity a
$[B]$	Matrix of damping terms in the finite element analysis
$F(\mathbf{X})$	Objective function
g_L	Lower-bound constraint
g_U	Upper-bound constraint
\tilde{g}_{jD}	Direct variable approximation for the j-th constraint
\tilde{g}_{jR}	Reciprocal variable approximation for the j-th constraint
$g_j(\mathbf{X})$	j-th constraint function
$\tilde{g}_j(\mathbf{X})$	Approximation of the j-th constraint function
$h_k(\mathbf{X})$	k-th equality constraint
$[I]$	Identity matrix
$[K]$	Stiffness matrix in the finite element analysis
$[K_d]$	Differential stiffness matrix
$[M]$	Mass matrix in the finite element analysis
p_j	j-th property value
p_j^U	Upper move limit on j-th property
p_j^1	j-th type-1 property (DVPREL1)
p_j^2	j-th type-2 property (DVPREL2)
p_j^L	Lower move limit on j-th property
$\{p^0\}$	Vector of initial property values
$\{P\}$	Load vector in the finite element analysis
r_j	j-th response value
r_j^L	Lower bound of the j-th structural response
r_j^U	Upper bound on the j-th structural response



r_j^1	j-th type-1 response (DRESP1)
r_j^2	j-th type-2 response (DRESP2)
r_j^3	j-th type-3 response (DRESP3)
$\{r\}$	Vector of analysis model responses (displacements, stresses, eigenvalue, etc.)
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
Y	Vector of intermediate variables used in formal approximations
x_i	i-th design variable
x_i^L	Lower-bound side constraint on i-th design variable
x_i^U	Upper-bound side constraint on i-th design variable
X	Vector of design variables
X^o	Initial value of vector of design variables (this characterizes the initial design).
X^*	Vector of design variables at the optimal design
$\{\quad\}^T$	Transpose of a vector
$[\quad]^T$	Transpose of a matrix
α	One-dimensional search parameter
α^*	Optimal value of one-dimensional search parameter for the current search direction 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	Adjoint solution vector
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
$\{\phi_n\}$	n-th eigenvector



$[\Phi]$	Modal transformation matrix
ρ	Mass density
ω	Frequency of oscillation of a dynamic system (rad/s)



Glossary of Terms

Active Constraint is a constraint whose numerical value is near zero. In MSC 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**.

Adjoint Sensitivity Analysis performs sensitivity analysis by first determining an adjoint load vector that is the change the response quantity due to a change in the solution vector. This can be compared with the **Direct Sensitivity Analysis** methods. The two methods give comparable sensitivity results but the efforts required in their computation can differ dramatically.

Analysis Discipline is used in Solution 200 to refer to the available analysis types: Statics, Normal Modes, Buckling, Direct and Modal Complex Eigenanalysis, Direct and Modal Frequency, Modal Transient, Static Aeroelasticity, divergence and Flutter.

Analysis Model defines the geometry, element connectivity, material properties, and loads associated with an MSC Nastran finite element analysis. The analysis model may be overridden by the design model.

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.

Approximation Concepts include design variable linking, constraint regionalization and deletion, and formal approximations. All of these approximations are available in MSC 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 [Equation \(1-1\)](#) through [Equation \(1-5\)](#).

Basis Vector is a collection of constant coefficients used to relate changes in structural properties to changes in design variables. 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**.

Beam Library contains standard cross section definitions for relating beam dimensions to beam properties. The MSC supplied section types can be augmented by user defined section types and all of the members of the beam library can be designed.

Beta Method is an optimization technique that minimizes an artificial design variable to minimize a maximum response. See [Acoustic Optimization](#).

Central Difference Sensitivities rely on finite difference perturbations to compute approximate derivatives, or sensitivities, of continuous functions. Central Difference Approximations, which perturb the design variables in both a forward and backward direction, yield greatly improved derivative approximations when compared to first order differencing schemes (forward and backward differences are first order.)

Continuous Design Variables can assume any value in a specified range and are to be distinguished from discrete design variables.

Constraint (Design Constraint) is defined as an inequality which must be satisfied in order to indicate a feasible design. MSC Nastran convention is that negative constraint values are deemed satisfied while positive values are violated.

Constraint Deletion is the process which 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 previously were deleted.

Constraint Regionalization refers to the grouping of constraints by type in preparation for constraint deletion. Regionalization is done in conjunction with constraint deletion to limit the number of constraints that need to be considered in a particular region of the structure.

Constraint Screening is an approximation concept which 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 which are temporarily deleted may be retained on subsequent design cycles.

Convergence is the term to indicate that the design process has reached a final state and can be terminated. This concept applies to both the **Design Cycle** and the **Design Iteration** processes. The converged design can be feasible or infeasible.



CSV (Comma Separated Values) File is a specialized output from design optimization which produces design optimization data in a comma separated values format that can be read into a spreadsheet and used to produce graphical displays of optimization results.

Dependent Design Variable is a design variable defined on a DESVAR entry which 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 which invokes the optimizer once for each design cycle. Within a design cycle a finite element analysis is performed, an approximate problem is created based on the design sensitivity analysis, 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 (or maximize) 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.

Designed Properties are those items that can be changed in the design optimization process. Properties can be those found on MSC Nastran property entries (such as the thickness of a plate), those found on material entries (such as Young's modulus) and those found on connectivity entries (such as a beam offset). Variations in shape are not considered Designed Properties.

Design Sensitivity Analysis is the process which 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 to 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. The range of the design variables is limited by upper and lower bounds on the variables and it is within this space that the optimizer searches for an optimal design while simultaneously trying to satisfy all of the imposed constraints.



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 nonzero 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 which 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.

Direct Sensitivity Analysis performs sensitivity analysis by first determining a pseudo load vector that is the sensitivity of the system matrix times solution vector. This can be compared with the Adjoint Sensitivity Analysis methods. The two methods give comparable sensitivity results but the efforts required in their computation can differ dramatically.

Discrete Design Variables are design variables that can only take on one of the specified discrete values, as opposed to continuous variables.

External Responses in MSC Nastran are those responses that are computed by an external process and included with the MSC Nastran optimization task either as they objective or by placing constraints on the response values.

Feasible Design is a design which satisfies all of the constraints. A feasible design may not be optimal.

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 [Equation \(2-28\)](#)) and central differences (see [Equation \(2-29\)](#)) are frequently used in MSC Nastran's semianalytic sensitivity analysis to provide low-cost derivative approximations.

First-level Response is a response that is directly available from an MSC Nastran analysis. These responses are identified for use in the design model with DRESP1 entries and are to be distinguished from second and third level responses.



Flat Design Space is a term used to designate the situation where significant changes in design variable values produce limited changes in the objective and constraint values. In this situation, the designer has latitude in selecting the final design parameters other than those produced by the computational optimization process without incurring a significant penalty.

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.

Forward (and Backward) Sensitivities perform finite difference sensitivity analysis by perturbing the design variable in a single direction. These methods are faster, but less accurate, than central differencing techniques.

Fully Stressed Design is a non-gradient based resizing algorithm that is used to resize element thicknesses or areas so as to produce a design where each designed property is subjected to its maximum allowable stress (or strain) under at least one loading condition.

Global Optimum is the term used to designate the optimum design that is the best among all positive relative minima. The design optimization process as implemented in MSC Nastran can lead to a relative, as opposed to, global optimum. This can be investigated by starting the design task from different points in the design space and comparing the optimal designs.

Gradient is defined as a vector of partial derivatives. See [Equation \(1-10\)](#). 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 optimizers in MSC Nastran are of this class.

Grid Perturbation refers to a small change in a grid point coordinate for a corresponding small 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 the 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 that is not expressed as a function of any other design variable. All design variables are independent unless they are made dependent by a DLINK entry. In design optimization, the optimizer varies the independent variables directly.

Infeasible Design is a design that violates one or more constraints.



Kuhn-Tucker Conditions provide the formal definition of an optimal design (see [The IPOPT Algorithm](#)). These conditions are implicitly evaluated in MSC Nastran in connection with convergence detection at the optimizer level.

Load Case Deletion represents a special case of constraint deletion where none of the constraints in a given subcase survive the constraint deletion process. In this circumstance, the direct method of sensitivity analysis can eliminate the load case from further consideration, thereby decreasing the amount of effort required in the sensitivity analysis.

Mode Tracking refers to the technique that is employed to allow the user to design a particular physical mode shape (such as the first torsion mode) rather than a specific mode number. The mode number for the physical mode can switch and the mode tracking permits the design task to switch as well.

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 [Move Limits](#)), move limits serve to restrict the search in this approximate design space to avoid numerical ill-conditioning due to poor approximations. Move limits are placed both on the design variables and on the designed properties.

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 2-18](#) in [Tests for Convergence](#), for a qualitative explanation of the situation. This is also referred to as a **Flat Design Space**.

Objective Function (Design Objective) is the function that the optimizer seeks to minimize (or maximize). 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 [Search Direction](#), and [Equation \(1-12\)](#).

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. By definition, the optimum design is feasible. It is possible that the optimum design is a relative minima 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, material properties and concentrated mass values 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 [Equation \(2-3\)](#).

Search Direction (Search Vector) an n -dimensional vector where n is the number of independent design variables, which 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 MSC 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. Physically, the variables can represent, for example, a radius, a coordinate location defining the center of a cutout or the percentage addition of taper to a control arm.

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 an upper or a lower bound on a design variable. Since the optimizer never searches outside of these bounds, these side constraints define the limits of the design space.

Soft Convergence refers to the design cycle convergence test that compares the design variables and properties that are output from an approximate optimization task with those used as input to approximate optimization task. This test, while not as conclusive as hard convergence, may be a suitable test to indicate that further redesign cycles are unlikely to produce an improved design.



Synthetic Response is another term for Second Level Response.

Third Level Responses are defined on DRESP3 Bulk Data entries and are also referred to as **External Responses**.

Type-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 MSC 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 MSC Nastran, the constant CTMIN (default = 0.003) is used to provide a numerical zero. This improves the numerical conditioning of the problem since trying to precisely satisfy an inequality relation is often a waste of computational resources.



References

1. Vanderplaats, G. N., *Multi-discipline Design Optimization*, Vanderplaats Research and Development, Inc., Colorado Springs, CO, 2009. Available on www.vrand.com
2. Haftka, R.T., Gurdal, Z. and Kamat, M.P. "Elements of Structural Optimization", *Springer - Science + Business*, 1990.
3. Vanderplaats, G. N., "ADS -- A FORTRAN Program for Automated Design Synthesis -- Version 1.10," *NASA Contractor Report 177985*, NASA Langley Research Center, Hampton, Virginia, 1985.
4. Nelson, R.B., "Simplified Calculations of Eigenvector Derivatives," *AIAA Journal*, Vol. 14, No. 9, pp. 1201-1205.
5. Belegundu, A.D. and Rajan, S.P., "Shape Optimal Design Using Isoparametric Elements," *Proceedings, 29th AIAA/ASME/ASEE/AMS/ASC Structures, Dynamics and Materials Conference*, pp. 696-701, Williamsburg, VA, April 1988.
6. Kohn, R.V. and Strang, G., "Optimal Design and Relaxation of Variational Problems", *Communications on Pure and Applied Mathematics*, 39, 1986, pp. 1-25 (Part I) 139-182 (Part II) 353-377 (Part III).
7. Ambrosio, L. and Buttazo, G., "An Optimal Design Problem with Perimeter Penalization", *Calculus of variations and Partial Differential Equations 1*, 1993, pp. 55-69.
8. Sigmund, O., *Design of Material Structures using Topology Optimization*, Ph.D. Thesis, Technical University of Denmark (DTU), Denmark, 1994.
9. Bendsoe, M.P. and Kikuchi, N., "Generating Optimal Topologies in Structural Design using a Homogenization Method", *Computer Methods in Applied Mechanics and Engineering*, 71, 1988, pp. 197-224.
10. Bendsoe, M.P., "Optimal Shape Design as a Material Distribution Problem", *Structural Optimization*, 1, 1989, pp. 193-202.
11. Xie, Y.M. and Steven, G.P., "A Simple Evolutionary Procedure for Structural Optimization", *Computers and Structures*, 49 (5), 1993, pp. 885-896.
12. Zhou, M. and Rozvany, G.I.N., "On the Validity of ESO type Methods in Topology Optimization", *Structural Multidisciplinary Optimization*, 21, 2001, pp. 80-83
13. Turner, M.J., "Design of Minimum Mass Structures with Specified Natural Frequencies", *AIAA Journal* Vol. 5, No. 3, March 1967.
14. Johnson, E.H. and Neill, D.J., "Automated Structural Optimization System (ASTROS)", *Volume III, Application Manual*, AFWAL TR-88-3028, December 1988, pp. 133-191.
15. Cassis, J.H., *Optimum Design of Structures Subjected to Dynamic Loads*, UCLA-ENG-7451, June 1974.
16. Peery, D.J. and Azar, J.J., *Aircraft Structures*, McGraw-Hill Book Company, 1982, pp 328-341.
17. M.K. Shin, K.J. Park and G.J. Park, "Optimization of Structures with Nonlinear Behavior Using Equivalent Loads," *Computer Methods in Applied Mechanics and Engineering*, 196 (2007) 1154–1167.



18. H. C. Gea and J. Luo, "Topology optimization of structures with geometric nonlinearities," *Computers & Structures* 79(2001) 1977-1985



B

Adding Your Own Beam Cross-Section Library

- Introduction 696
- BSCON Subroutine 698
- BSBRPD Subroutine 700
- BSGRQ Subroutine 711
- BSBRT Subroutine 712
- BSBRID Subroutine 715
- BSBRGD Subroutine 721
- BSBRCD Subroutine 727
- BSMG Subroutine 733
- Linking Your Library to MSC Nastran 734
- Example of Building and Linking a Beam Server 735



Introduction

This section details how to add your own cross section type that would be in addition to the 19 standard section types that MSC provides. As this section shows, adding this section is an involved process that requires the coding and connection of a number of subroutines. Before embarking on using this capability, it is recommended that you explore the PBRSECT (for bars) and PBMSECT (for beams) bulk data entry that allows you to define a general cross section using a set of points and instructions. Some considerations are:

- a. The PBRSECT/PBMSECT does not require the developments of this appendix in terms of coding, building and linking of the beam server.
- b. Basic design of the PBRSECT/PBMSECT is supported (see Special Considerations when Designing One-Dimensional Bending Elements).
- c. If there is a beam cross section type that is not supported by one of the standard sections that is needed extensively in a design and the dimensions of the sections are designable, there may be a payoff in the methods of this appendix.
- d. The PBEAML supports tapered sections while the PBMSECT has a constant cross section.

The standard cross sections provided by MSC should be adequate in the majority of cases. If these standard sections are not adequate for your purposes, you can add your own library of cross sections to suit your needs. To add your own library, you need to write a few simple subroutines in FORTRAN and link them to MSC Nastran through inter-process communications. This process requires writing and/or modifying up to eight basic subroutines.

1. BSCON -- Defines the number of dimensions for each of the section types.
2. BSBRPD -- Calculates section properties based on section dimensions.
3. BSGRQ -- Defines NSECT, the number of section types, and NDIMAX, the maximum number of dimensions (including nonstructural mass) required by any of the sections.
4. BSBRT -- Provides the name, number of dimensions and number of design constraints for each section type.
5. BSBRID -- Provides information for the calculation of gradients of section properties with respect to section dimensions.
6. BSBRGD -- Calculates any nonlinear gradients of section properties with respect to section dimensions.
7. BSBRCD -- Defines constraints in the design of section dimensions.
8. BSMSG -- A utility routine; handles errors that occur in the beam library.

BSCON and BSBRPD are always required. BSGRQ, BSBRT, and BSBRID are required if you wish to perform sensitivity and/or optimization tasks using the beam library. BSBRGD is required if you are providing nonlinear analytical sensitivities in the design task, and BSBRCD is an optional routine that can be provided to help the optimizer to stay within physical design constraints. BSMSG handles any error messages you feel are appropriate.



This appendix describes each of these basic routines in turn and provides MSC Software's version of each. Routines that are called by these basic routines are also described with adequate examples to allow you to construct your own library. All the example routines shown are for the 32-bit machines.

For 64-bit machines, all the routine names that end with "D" should be changed to end with "S", and all real variables must be designated as single precision instead of double precision. Therefore, the naming convention for routines on 64-bit machines is BSCON, BSBRPS, BSGRQ, BSBRT, BSPRIS, BSBRGS, BSBRCs, and BSMSG.



BSCON Subroutine

This routine provides the number of fields in the continuation lines to be read from the Bulk Data entries PBARL and PBEAML for each cross section in the library. The value of the ENTYP variable may be 0, 1, or 2. When ENTYP=0, the value returned is the number of DIMi. When ENTYP=1, the value returned includes both the DIMi and NSM fields. The value of 1 is used for PBARL only. When ENTYP = 2, the value returned includes the DIMi, NSM, SO, and XIXB fields for 11 different stations. The value of 2 applies to PBEAML only.

The calling sequence and example routine for the standard MSC library is given below.

Listing B-1 BSCON Subroutines

```

SUBROUTINE BSCON (GRPID,TYPE,ENTYP,NDIMI,ERROR)
C
C -----
C      Purpose
C          To get the dimension information for 'MSC' types
C
C      Arguments:
C
C          GRPID    input INTEGER INTEGER ID OF THIS GROUP OR GROUP NAME.
C          TYPE     input CHARACTER*8 TYPE OF CROSS SECTION
C          ENTYP    input INTEGER - 1: PBARL, 2:PBEAML
C          NDIMI    output NUMBER OF INPUT FIELDS FOR THE 'ENTYP' CROSS SECTION
C
C      Called by BCCON
C -----
C          IMPLICIT INTEGER (I-N)
C          IMPLICIT REAL (A-H,O-Z)
C
C          INTEGER      ENTYP,GRPID,ERROR
C          CHARACTER*8   TYPE
C
C      Local variables
C          INTEGER      NAM(2)
C          DATA NAM/4HBSCO,4HN    /
C
C      Default to 'nothing wrong'
C      ERROR = 0
C
C      Dimensions vary with section type
C          IF      ( TYPE.EQ.'ROD'      ') THEN
C              NDIMI = 1
C          ELSEIF( TYPE.EQ.'TUBE'      ' .OR. TYPE.EQ.'BAR'      ') THEN
C              NDIMI = 2
C          ELSEIF( TYPE.EQ.'HEXA'      ') THEN
C              NDIMI = 3
C          ELSEIF( TYPE.EQ.'BOX'      ' .OR. TYPE.EQ.'T'      ' .OR.
C+           TYPE.EQ.'L'      ' .OR.
C+           TYPE.EQ.'CHAN'    ' .OR. TYPE.EQ.'CROSS'    ' .OR.
C+           TYPE.EQ.'H'      ' .OR.
C+           TYPE.EQ.'I1'     ' .OR. TYPE.EQ.'T1'      ' .OR.
C+           TYPE.EQ.'CHAN1'  ' .OR. TYPE.EQ.'Z'      ' .OR.
C+           TYPE.EQ.'T2'     ' .OR. TYPE.EQ.'CHAN2'  ' .OR.
C+           TYPE.EQ.'HAT'    ' .OR. TYPE.EQ.'HAT'    ' ) THEN
C              NDIMI = 4
C          ELSEIF( TYPE.EQ.'HAT1'    ') THEN
C              NDIMI = 5
C          ELSEIF( TYPE.EQ.'I'      ' .OR. TYPE.EQ.'BOX1'    ') THEN
C              NDIMI = 6
C          ELSE
C              Is the return here necessary ?

```



```
        ERROR = 5150
        RETURN
    ENDIF

C      Specify ?
IF (ENTYP.EQ.1) NDIMI = NDIMI+1
IF (ENTYP.EQ.2) NDIMI = (NDIMI+3)*11

C All done
RETURN
END
```



BSBRPD Subroutine

Finite element analysis requires section properties such as area, moment of inertia, etc., instead of section dimensions. Therefore, the dimensions specified on PBARL and PBEAML need to be converted to the equivalent properties specified on PBAR and PBEAM entries. The images of all these entries are stored in EPT datablock as records.

The BSBRPD subroutine is the interface of your properties evaluator with MSC Nastran. You may use your own naming convention for the subroutines that calculate the cross-section properties from the dimensions. The calling tree used for the MSC standard library is shown in [Figure B-1](#).

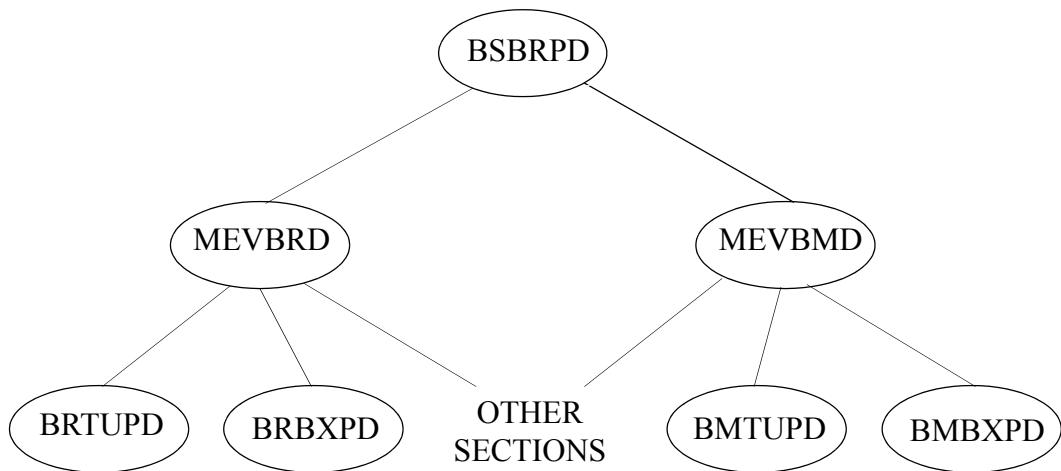


Figure B-1 Calling Tree Generate Property Data

The BSBRPD calls the bar evaluator routine MEVBRD for the BAR element and the beam evaluator routine MEVBMD for the BEAM element. The evaluators in turn call the routines for each section. The routines are named BRXXPD and BMXXPD, where XX is a two-letter identifier for the section. For example, the routines for the TUBE section are called BRTUPD and BMTUPD. The details for various routines are given in [Listing B-2](#).

Listing B-2 BSBRPD Subroutines

```

SUBROUTINE BSBRPD (GRPID,ENTYP,TYPE,IDI,NID,IDO,NIDO,DIMI,NDIMI,
+                   DIMO,NDIMO,ERROR)
C=====
C   PURPOSE:
C       1.SET INTEGER PART
C       2.DETECT HOW MANY SECTIONS
C       3.UNIFORM OR LINEAR BEAM
C       4.CHECK WHAT ENTITY TYPE'S SUBROUTINE TO BE CALLED
C          (1:PBARL ;2:PBEAML)
C-----
C   CALLED BY:
C       BCBRP
C-----
C   CALLS:
C       MEVBRD ,MEVBMD
  
```



```

C-----
C   IMPLICIT DECLARATIONS
      IMPLICIT INTEGER (I-N)
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
C-----
C   EXPLICIT DECLARATIONS
      INTEGER ENTP, ERROR, GRPID
      CHARACTER*8 TYPE
C-----
C   LOCAL VARIABLES
      INTEGER NAM(2)
C-----
C   DIMENSION STATEMENTS
      INTEGER IDI(NID), IDO(NIDO)
      DOUBLE PRECISION DIMI (NDIMI), DIMO (NDIMO)
C-----
C   DATA
      DATA NAM/4HBSBR,4HPD /
C=====
C== ENTP=1, FOR PBARL; 2, FOR PBEAML
      IF (ENTP.EQ.1) THEN
          CALL MEVBRD(GRPID,TYPE,IDI,NID,IDO,NIDO,DIMI,NDIMI,DIMO,NDIMO
          + ,ERROR)
      END IF
      IF (ENTP.EQ.2) THEN
          CALL MEVBMD(GRPID,TYPE,IDI,NID,IDO,NIDO,DIMI,NDIMI,DIMO,NDIMO
          + ,ERROR)
      END IF
C-----
      RETURN
END

```

MEVBRD and MEVBMD Subroutines

MEVBRD and MEVBMD are the brancher routines for the various sections, and convert the section dimensions to section properties for BAR and BEAM elements. You may rename these routines as you like or move the function of these routines to BSBRPD. These routines call the BRXXPD or BMXXPD routines where XX is the two-letter keyword for various section types. The MEVBRD routine for the MSC standard library is given in [Listing B-3](#). The MEVBMD routine is not shown but can be developed by using the BMXXPD convention.

Listing B-3 MEVBRD Subroutine

```

SUBROUTINE MEVBRD(GRPID,TYPE, ID,NID,IDO,NIDO,DIMI,NDIMI,DIMO,
+                   NDIMO,ERROR)
C=====
C   Purpose
C           Call the default type subroutine to convert PBARL to PBAR
C
C   Arguments
C
C   GRPID input    int     ID of group
C   TYPE    input    char    Type of cross section
C   DIMI   input    flt     Dimension values of cross section
C   NDIMI  input    int     Size of DIMI array
C   DIMO   output   flt     Properties of cross section
C   NDIMO  output   flt     Size of DIMO array
C   ERROR  output   int     Type of error
C
C   Method
C           Call the subroutine with respect to the section type
C
C   Called by

```



```

C           BSBRPD
C
C   CALLS      BRRDPD,BRTUPD,BRBRPD,BRBXPD,BRIIPD,BRTTPD,BRLLPD,BRCHPD
C-----
C           IMPLICIT INTEGER (I-N)
C           IMPLICIT DOUBLE PRECISION (A-H, O-Z)
C
C   Calling sequence arguments
C   INTEGER          ERROR,GRPID, ID(NID), IDO(NIDO)
C   CHARACTER*8       TYPE
C   DOUBLE PRECISION DIMI (NDIMI), DIMO (NDIMO)
C
C   Local variables
C   INTEGER NAM(2)
C
C   Local data
C   DATA NAM/4HMEVB,4HRD  /
C=====
C   Clear the output array before usage
C   CALL      ZEROD ( DIMO, NDIMO )
C   CALL      ZEROI ( IDO, NIDO  )
C   ERROR = 0
C
C   IF      ( TYPE.EQ.'ROD      ') THEN
C           CALL BRRDPD(ID,NID,IDO,NIDO,DIMI,NDIMI,DIMO,NDIMO,ERROR)
C   ELSEIF( TYPE.EQ.'TUBE     ') THEN
C           CALL BRTUPD(ID,NID,IDO,NIDO,DIMI,NDIMI,DIMO,NDIMO,ERROR)
C   ELSEIF( TYPE.EQ.'BAR      ') THEN
C           CALL BRBRPD(ID,NID,IDO,NIDO,DIMI,NDIMI,DIMO,NDIMO,ERROR)
C   ELSEIF( TYPE.EQ.'BOX      ') THEN
C           CALL BRBXPD(ID,NID,IDO,NIDO,DIMI,NDIMI,DIMO,NDIMO,ERROR)
C   ELSEIF( TYPE.EQ.'I        ') THEN
C           CALL BRIIPD(ID,NID,IDO,NIDO,DIMI,NDIMI,DIMO,NDIMO,ERROR)
C   ELSEIF( TYPE.EQ.'T        ') THEN
C           CALL BRTTPD(ID,NID,IDO,NIDO,DIMI,NDIMI,DIMO,NDIMO,ERROR)
c10939ELSEIF( TYPE.EQ.'L      ') THEN
c10939  CALL BRLLPD(ID,NID,IDO,NIDO,DIMI,NDIMI,DIMO,NDIMO,ERROR)
C   ELSEIF( TYPE.EQ.'CHAN    ') THEN
C           CALL BRCHPD(ID,NID,IDO,NIDO,DIMI,NDIMI,DIMO,NDIMO,ERROR)
C   ELSEIF( TYPE.EQ.'CROSS   ') THEN
C           CALL BRCRPD(ID,NID,IDO,NIDO,DIMI,NDIMI,DIMO,NDIMO,ERROR)
C   ELSEIF( TYPE.EQ.'H        ') THEN
C           CALL BRHHPD(ID,NID,IDO,NIDO,DIMI,NDIMI,DIMO,NDIMO,ERROR)
C   ELSEIF( TYPE.EQ.'T1      ') THEN
C           CALL BRT1PD(ID,NID,IDO,NIDO,DIMI,NDIMI,DIMO,NDIMO,ERROR)
C   ELSEIF( TYPE.EQ.'I1      ') THEN
C           CALL BRI1PD(ID,NID,IDO,NIDO,DIMI,NDIMI,DIMO,NDIMO,ERROR)
C   ELSEIF( TYPE.EQ.'CHAN1  ') THEN
C           CALL BRC1PD(ID,NID,IDO,NIDO,DIMI,NDIMI,DIMO,NDIMO,ERROR)
C   ELSEIF( TYPE.EQ.'Z        ') THEN
C           CALL BRZZPD(ID,NID,IDO,NIDO,DIMI,NDIMI,DIMO,NDIMO,ERROR)
C   ELSEIF( TYPE.EQ.'CHAN2  ') THEN
C           CALL BRC2PD(ID,NID,IDO,NIDO,DIMI,NDIMI,DIMO,NDIMO,ERROR)
C   ELSEIF( TYPE.EQ.'T2      ') THEN
C           CALL BRT2PD(ID,NID,IDO,NIDO,DIMI,NDIMI,DIMO,NDIMO,ERROR)
C   ELSEIF( TYPE.EQ.'BOX1   ') THEN
C           CALL BRB1PD(ID,NID,IDO,NIDO,DIMI,NDIMI,DIMO,NDIMO,ERROR)
C   ELSEIF( TYPE.EQ.'HEXA   ') THEN
C           CALL BRHXPD(ID,NID,IDO,NIDO,DIMI,NDIMI,DIMO,NDIMO,ERROR)
C   ELSEIF( TYPE.EQ.'HAT    ') THEN
C           CALL BRHTPD(ID,NID,IDO,NIDO,DIMI,NDIMI,DIMO,NDIMO,ERROR)
C   ELSEIF( TYPE.EQ.'HAT1   ') THEN
C           CALL BRCLPD(ID,NID,IDO,NIDO,DIMI,NDIMI,DIMO,NDIMO,ERROR)
C   ELSE
C           ERROR = 5150
END IF

```



```
C-----
      RETURN
      END
```

BRXXPD and BMXXPD Subroutines

The purpose of the BRXXPD and BMXXPD routines is to calculate the properties from the section dimensions. For each cross section, subroutines are required to convert the images of PBARL and PBEAML records to the images of PBAR and PBEAM records in the EPT datablock.

BRTUPD Subroutine

BRTUPD is an example routine that shows how to convert PBARL EPT record to PBAR EPT record for the Tube section. First, the details for the PBARL and PBAR record are shown, and then the routine itself is given.

PBARL Record

The PBARL record in the EPT datablock is derived from the PBARL Bulk Data entry and is given in [Table B-1](#).

Table B-1 PBARL (9102, 91, 52)

Word	Name	Type	Description
1	PID	I	Property identification number
2	MID	I	Material identification number
3	Group	Char	Group Name
4	Group	Char	Group Name
5	TYPE	Char4	Cross-section Type
6	TYPE	Char4	Cross-section Type
7	Dim1	RS	Dimension1
8	Dim2	RS	Dimension2
n+7-1	Dim n	RS	Dimension n (note that the final dimension is the nonstructural mass)
n+7	Flag	I	-1. Flag indicating end of cross-section dimensions

PBAR Record

The PBAR record in the EPT datablock is derived from the PBAR Bulk Data entry and consists of 19 words. It is a replica of the Bulk Data entry, starting with PID field. The word 8 in the record is set to 0.0 since the field 9 in the first line of the PBAR Bulk Data entry is not used. The details of the PBAR record are given in [Table B-2](#).



Table B-2 PBAR (52, 20, 181)

Word	Name	Type	Description
1	PID	I	Property identification number.
2	MID	I	Material identification number.
3	A	RS	Area of cross-section.
4	I1	RS	Area moment of inertia for bending in plane 1.
5	I2	RS	Area moment of inertia for bending in plane 2.
6	J	RS	Torsional constant.
7	NSM	RS	Nonstructural mass per unit length.
8	FE	RS	Not used. Set of 0.0.
9	C1	RS	Stress recovery location.
10	C2	RS	Stress recovery location.
11	D1	RS	Stress recovery location.
12	D2	RS	Stress recovery location.
13	E1	RS	Stress recovery location.
14	E2	RS	Stress recovery location.
15	F1	TS	Stress recovery location.
16	F2	RS	Stress recovery location.
17	K1	RS	Area factor of shear for plane 1.
18	K2	RS	Area factor of shear for plane 2.
19	I12	RS	Area product of inertia.

Listing B-4 BRTUPD Subroutine

```

SUBROUTINE BRTUPD(ID,NID,IDO,NIDO,DIMI,NDIMI,DIMO,NDIMO,ERROR)
C=====
C   Purpose:
C       Convert PBARL(entity type : TUBE) to PBAR
C
C   Arguments:
C
C     ID      input  int Array of values PID, MID contained in PBARL
C             entries
C     NID    input  int Size of ID array, NID=2 for PBARL entry
C     IDO    output int Array of integer values contained in PBAR
C             entries
C     NIDO   output int Size of IDO array, NIDO=2 for PBAR entry
C     DIMI   input  flt Array of dimension values for cross-section
C             (SEE FIG. 5 IN MEMO SSW-25, REV. 4, DATE 8/16/94)
C     NDIM   input  int Size of DIMI array
C     DIMO   output flt Array of property values for cross-section
C     NDIMO  output int Size of DIMO array
C     ERROR  output int Type of error
C
C   DESCRIPTION FOR DIMO ARRAY:

```



```

C      DIMO (1) = A
C      DIMO (2) = I1
C      DIMO (3) = I2
C      DIMO (4) = J
C      DIMO (5) = NSM
C      DIMO (6) = FE
C      DIMO (7) = C1
C      DIMO (8) = C2
C      DIMO (9) = D1
C      DIMO (10) = D2
C      DIMO (11) = E1
C      DIMO (12) = E2
C      DIMO (13) = F1
C      DIMO (14) = F2
C      DIMO (15) = K1
C      DIMO (16) = K2
C      DIMO (17) = I12
C
C      Method
C
C      Called by:
C          MEVBRD
C-----
C          IMPLICIT INTEGER (I-N)
C          IMPLICIT DOUBLE PRECISION (A-H,O-Z)
C
C      Calling sequence arguments
C          INTEGER ID(NID), IDO(NIDO), ERROR
C          DOUBLE PRECISION DIMI(NDIMI), DIMO(NDIMO)
C
C      Local variables
C          INTEGER NAM(2)
C
C      Nastran common blocks
C          COMMON /CONDAD/PI
C
C      Local data
C          DATA NAM/4HBRTU,4HPD /
C=====
C
C      === WRITE THE PART OF INTEGER
C          DO 30 II = 1,NID
C              IDO(II) = ID(II)
C 30      CONTINUE
C
C          DIM1 = DIMI(1)
C          DIM2 = DIMI(2)
C          DIMO(1) = PI*(DIM1*DIM1-DIM2*DIM2)
C          DIMO(2) = PI*(DIM1**4-DIM2**4)/4.D0
C          DIMO(3) = DIMO(2)
C          DIMO(4) = PI*(DIM1**4-DIM2**4)/2.D0
C          DIMO(5) = DIMI(3)
C          DIMO(6) = 0.D0
C          DIMO(7) = DIM1
C          DIMO(8) = 0.D0
C          DIMO(9) = 0.D0
C          DIMO(10) = DIM1
C          DIMO(11) = -DIM1
C          DIMO(12) = 0.D0
C          DIMO(13) = 0.D0
C          DIMO(14) = -DIM1
C          DIMO(15) = 0.5D0
C          DIMO(16) = 0.5D0
C          DIMO(17) = 0.D0
C
C          IF ( DIMI(1).LE.DIMI(2) ) error = 5102
C-----

```



```
RETURN
END
```

BMTUPD Subroutine

BMTUPD is an example routine that shows how to convert PBEAM EPT record to PBEAM EPT record for the Tube section. First, the details of the PBEAML and the PBEAM records are shown, and then the routine itself is given.

PBEAML Record

The PBEAML record on the EPT datablock is a derived from the PBEAML Bulk Data entry and is given in [Table B-3](#).

Table B-3 PBEAML (9202, 92, 53)

Word	Name	Type	Description
1	PID	I	Property ID.
2	MID	I	Material ID.
3		Char	Group Name.
4		Char	Group Name.
5		Char4	Cross-section Type.
6		Char4	Cross-section Type.
7		RS	Stress output request flag, 1=yes, =no.
8		RS	X/XB - parametric location of the station.
9		RS	Dimension 1.
10		RS	Dimension 2.
n+9-1		RS	Dimension n (note that the final dimension is the nonstructural mass).
n+9		I	-1. Flag indicating end of cross-section dimensions.
Words 7 through 11 repeat 10 times.			

PBEAM Record

The PBEAM record in EPT datablock consists of 197 words. The first five words and the last 16 words are common to all the 11 stations. Each of the 11 stations has its own 16 unique words. The details of the PBEAM record are given in [Table B-4](#).



Table B-4 PBEAM (5402, 54, 262)

Word	Name	Type	Description
1	PID	I	Property identification number.
2	MID	I	Material identification number.
3	N	I	Number of intermediate stations.
4	CCF	I	Constant cross-section flag. 1 = constant, 2 = variable.
5	X	RS	Unused.
6	SO	RS	Stress output request. 1.0 = yes, 0.0 = no.
7	XXB	RS	Parametric location of the station. Varies between 0. and 1.0.
8	A	RS	Area.
9	I1	RS	Moment of inertia for bending in plane 1.
10	I2	RS	Moment of inertia for bending in plane 2.
11	I12	RS	Area product of inertia.
12	J	RS	Torsional constant.
13	NSM	RS	Nonstructural mass.
14	C1	RS	Stress recovery location.
15	C2	RS	Stress recovery location.
16	D1	RS	Stress recovery location.
17	D2	RS	Stress recovery location.
18	E1	RS	Stress recovery location.
19	E2	RS	Stress recovery location.
20	F1	RS	Stress recovery location.
21	F2	RS	Stress recovery location.
Words 6 through 21 repeat 10 times.			
182	K1	RS	Area factor for shear for plane 1.
183	K2	RS	Area factor for shear for plane 2.
184	S1	RS	Shear-relief coefficient for plane 1.
185	S2	RS	Shear-relief coefficient for plane 2.
186	NSIA	RS	Nonstructural mass moment of inertia at end A.
187	NSIB	RS	Nonstructural mass moment of inertia at end B.
188	CWA	RS	Warping coefficient for end A.
189	CWB	RS	Warping coefficient for end B.



Table B-4 PBEAM (5402, 54, 262)

Word	Name	Type	Description
190	M1A	RS	Y-coordinate of center of gravity for nonstructural mass at end A.
191	M2A	RS	Z-coordinate of center of gravity for nonstructural mass at end A.
192	M1B	RS	Y-coordinate of center of gravity for nonstructural mass at end B.
193	M2B	RS	Z-coordinate of center of gravity for nonstructural mass at end B.
194	N1A	RS	Y-coordinate for neutral axis at end A.
195	N2A	RS	Z-coordinate for neutral axis at end A.
196	N1B	RS	Y-coordinate for neutral axis at end B.
197	N2B	RS	Z-coordinate for neutral axis at end B.

Listing B-5 BMTUPD Subroutine.

```

SUBROUTINE BMTUPD(ID,NID,IDO,NIDO,DIMI,NDIMI,DIMO,NDIMO,ERROR)
C=====
C   Purpose
C      Convert PBEAML(entity type : TUBE) to PBEAM
C
C   Arguments
C
C   ID      input  int  Contain the integer information PID, MID
C   NID     input  int  Size of ID array, NID = 2
C   DIMI    input  flt  Dimension values of cross section
C                  ( See FIG.5 IN MEMO SSW-25, REV. 4, DATE 8/16/94)
C   NDIMI   input  int  Size of DIMI array
C   IDO     output int  Contain the integer information PID,MID,N,CCF
C   NIDO    output int  Size of IDO array, NIDO = 4
C   DIMO    output flt  Properties of cross section
C   NDIMO   output int  Size of DIMO array
C   ERROR   output int  Type of error
C
C   Description for DIMO array
C   DIMO (1)  = X
C   DIMO (2)  = SO
C   DIMO (3)  = XXB
C   DIMO (4)  = A
C   DIMO (5)  = I1
C   DIMO (6)  = I2
C   DIMO (7)  = I12
C   DIMO (8)  = J
C   DIMO (9)  = NSM
C   DIMO (10) = C1
C   DIMO (11) = C2
C   DIMO (12) = D1
C   DIMO (13) = D2
C   DIMO (14) = E1
C   DIMO (15) = E2
C   DIMO (16) = F1
C   DIMO (17) = F2
C   DIMO(2) thru DIMO(17) repeat 11 times
C   DIMO (178) = K1

```



```

C      DIMO (179) = K2
C      DIMO (180) = S1
C      DIMO (181) = S2
C      DIMO (182) = NSIA
C      DIMO (183) = NSIB
C      DIMO (184) = CWA
C      DIMO (185) = CWB
C      DIMO (186) = M1A
C      DIMO (187) = M2A
C      DIMO (188) = M1B
C      DIMO (189) = M2B
C      DIMO (190) = N1A
C      DIMO (191) = N2A
C      DIMO (192) = N1B
C      DIMO (193) = N2B
C
C      Method
C                  Simply calculate the properties and locate that data
C                  to the image of PBEAM entries
C      Called by
C                  MEVBMD
C-----
C      IMPLICIT INTEGER (I-N)
C      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
C
C      Calling sequence arguments
C      INTEGER           ERROR, ID(NID), IDO(NIDO)
C      DOUBLE PRECISION DIMI(NDIMI), DIMO(NDIMO)
C
C      Local variables
C      INTEGER NAM(2)
C
C      NASTRAN common blocks
C      COMMON /CONDAD/PI
C
C      Local data
C      DATA NAM/4HBMTU,4HPD /
C=====
C
C==== WRITE THE PART OF INTEGER
C      DO 30 II = 1,NID
C          IDO(II) = ID(II)
C 30    CONTINUE
C-----
C==== DETECT HOW MANY STATION , CONSTANT OR LINEAR BEAM.
C-----
C      ISTATC = NDIMI/11
C      DO 35 II = 0,10
C          NW = II*ISTATC
C          IF (DIMI(3+NW).EQ.0.D0) THEN
C              call bmsido( ido(3), ido(4),ii)
C              GO TO 40
C          END IF
C 35    CONTINUE
C      If we got here , we have all the stations, so set the values
C
C      if( dimi(2+9*istatc).eq. dimi(2+10*istatc) ) then
C          ii = 9
C      else
C          ii = 10
C      endif
C      ido(3) = ii
C      ido(4) = 0
C
C 40    continue
C      DIMO(1) = 0.D0
C      DO 100 L1 = 0,10
C          LC = 16*L1

```



```
NW = L1*ISTATC
IF (DIMI(3+NW+ISTATC).EQ.0.D0) LC = 160
DIM1 = DIMI(3+NW)
DIM2 = DIMI(4+NW)

IF ( DIM1.LE.DIM2 ) ERROR = 5102

DIMO(2+LC) = DIMI(1+NW)
DIMO(3+LC) = DIMI(2+NW)
DIMO(4+LC) = PI*(DIMI*DIM1-DIM2*DIM2)
DIMO(5+LC) = PI*(DIMI**4-DIM2**4)/4.D0
DIMO(6+LC) = DIMO(5+LC)
DIMO(7+LC) = 0.D0
DIMO(8+LC) = PI*(DIMI**4-DIM2**4)/2.D0
DIMO(9+LC) = DIMI(5+NW)
DIMO(10+LC) = DIM1
DIMO(11+LC) = 0.D0
DIMO(12+LC) = 0.D0
DIMO(13+LC) = DIM1
DIMO(14+LC) = -DIM1
DIMO(15+LC) = 0.D0
DIMO(16+LC) = 0.D0
DIMO(17+LC) = -DIM1
IF (LC.EQ.160) GO TO 110
100 CONTINUE
110 DIMO(178) = 0.5D0
DIMO(179) = 0.5D0
C-----
300 RETURN
END
```



BSGRQ Subroutine

For the optimization of PBARL or PBEAML entries, you need to provide overall information on the number of section types in your library and the maximum number of fields on the continuation lines. For the PBARL, the maximum number is the maximum number of dimensions plus 1 for the non-structural mass field. For the PBEAML, it is the number of words per station times the eleven possible stations. The count for the number of station words is the maximum number of dimensions plus three additional words for SO,X/XB and NSM. BSGRQ is only required if you wish to perform sensitivity or optimization with the section dimensions. The calling sequence and example subroutine for the MSCBML1 library is shown in [Listing B-6](#). It is seen that NDIMAX is set to 7 for the PBARL and 99 for the PBEAML.

Listing B-6 BSGRQ Routine

```

SUBROUTINE BSGRQ (GRPID,ENTYP, NSECT,NDIMAX,ERROR)
C
C =====
C PURPOSE:
C      GET DIMENSION INFORMATION OF BEAM CROSS SECTION OF MSCBML0 TYPES
C -----
C ARGUMENTS:
C
C   INPUT      :
C     GRPID    - GROUP NAME
C     ENTYP     - INTEGER - Entity Type.
C                  (1: PBARL, 2:PBEAML )
C   OUTPUT      :
C     NSECT     - NUMBER OF DIFFERENT SECTION TYPES
C     NDIMAX    - MAXIMUM NUMBER OF DIMENSION FOR ANY SECTION TYPE
C     ERROR      - INDICATES IF AN ERROR HAS OCCURRED. THE CODE RETURNED
C                  INDICATES THE TYPE OF ERROR
C
C   REASSIGNED :
C -----
C CALLED BY:
C          BCGRQ routine
C -----
C
      integer entyp
C
      if ( entyp .eq. 1 ) then
        nsect = 21
        ndimax = 7
      else if ( entyp .eq. 2 ) then
        nsect = 21
        ndimax = 99
      endif
C
      RETURN
END

```

Note: The arguments GRPID and ERROR are not used. GRPID is reserved for future use. You may use the ERROR argument to send an error code which could later be used to print an error message.



BSBRT Subroutine

The BSBRT routine provides the name, number of dimension, and the number of constraints for each section in the library. As an example, the name of the tube section, shown in [Figure B-2](#), in the MSC standard library is “TUBE”, the number of dimension for the tube section is three (OUTER RADIUS, INNER RADIUS, and NSM) and there is one physical constraint. This subroutine is shared by the PBARL and PBEAML. For the PBEAML, the numbers apply to each station.

The physical constraint is that the inner radius (DIM2) cannot be greater than outer radius (DMI1). It is necessary to specify the constraints so that the optimization of the section dimension in SOL 200 does not result into an inconsistent shape.

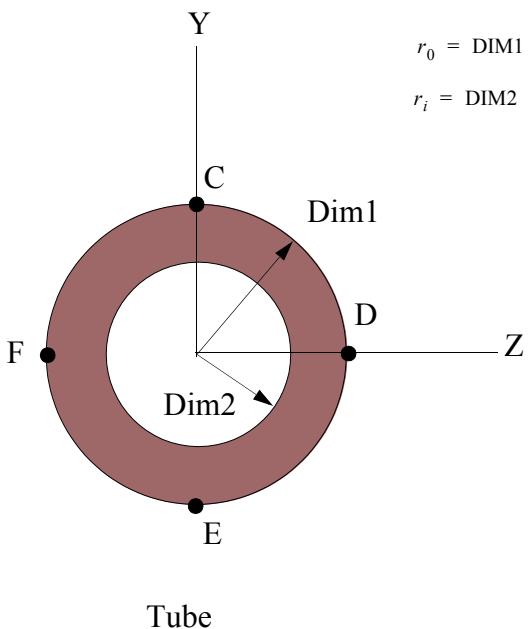


Figure B-2 Tube Section

This routine is called during the optimization process. The calling sequence and example routine is given in [Listing B-7](#).

Listing B-7 BSBRT Subroutine

```
SUBROUTINE BSBRT(GRPID,ENTYP,TYPE,NDIM,NCONST,NSECT,ERROR)
C
C =====
C Purpose
C      Get type dimensioning information for default sections
C
C Arguments
C
C      GRPID   input integer      The ID of group name
C      ENTYP   input integer      1: PBARL, 2: PBEAML
```



```

C      TYPE    output character*8  Arrays for cross section types
C      NDIM    output integer     Number of dimensions for isect
C      NCONST  output integer     section type
C      NCONST  output integer     Number of dimensional constraints
C      imposed by isect section type
C      NSECT   input  integer    Number of sections
C      ERROR   output integer    Type of error
C
C      Method
C          Transfers dimension information for various sections
C
C      Called by
C          BCBRT routine
C -----
C          IMPLICIT INTEGER (I-N)
C          IMPLICIT DOUBLE PRECISION (A-H,O-Z)
C
C      Calling sequence arguments
C      INTEGER      GRPID,ENTYP,NDIM(NSECT),NCONST(NSECT)
C      CHARACTER*8  TYPE(NSECT)
C
C      Local variables
C      INTEGER NAM(2)
C
C      Local data
C      DATA NAM/4HBSBR,4HT    /
C =====
C
C          GRPID = 1
C          ENTYP = 1
C
C          TYPE(1) = 'ROD'
C          NDIM(1) = 2
C          NCONST(1) = 0
C
C          TYPE(2) = 'TUBE'
C          NDIM(2) = 3
C          NCONST(2) = 1
C
C          TYPE(3) = 'BAR'
C          NDIM(3) = 3
C          NCONST(3) = 0
C
C          TYPE(4) = 'BOX'
C          NDIM(4) = 5
C          NCONST(4) = 2
C
C          TYPE(5) = 'I'
C          NDIM(5) = 7
C          NCONST(5) = 3
C
C          TYPE(6) = 'T'
C          NDIM(6) = 5
C          NCONST(6) = 2
C
C          TYPE(7) = 'L'
C          NDIM(7) = 5
C          NCONST(7) = 2
C
C          TYPE(8) = 'CHAN'
C          NDIM(8) = 5
C          NCONST(8) = 2
C
C          TYPE(9) = 'CROSS'
C          NDIM(9) = 5
C          NCONST(9) = 1
C
C          TYPE(10) = 'H'

```



```
NDIM(10) = 5
NCONST(10) = 1
C
TYPE(11) = 'T1'
NDIM(11) = 5
NCONST(11) = 1
C
TYPE(12) = 'I1'
NDIM(12) = 5
NCONST(12) = 1
C
TYPE(13) = 'CHAN1'
NDIM(13) = 5
NCONST(13) = 1
C
TYPE(14) = 'Z'
NDIM(14) = 5
NCONST(14) = 1
C
TYPE(15) = 'CHAN2'
NDIM(15) = 5
NCONST(15) = 2
C
TYPE(16) = 'T2'
NDIM(16) = 5
NCONST(16) = 2
C
TYPE(17) = 'BOX1'
NDIM(17) = 7
NCONST(17) = 2
C
TYPE(18) = 'HEXA'
NDIM(18) = 4
NCONST(18) = 1
C
TYPE(19) = 'HAT'
NDIM(19) = 5
NCONST(19) = 2
C
TYPE(20) = 'HAT1'
NDIM(20) = 6
NCONST(20) = 3
C
RETURN
END
```



BSBRID Subroutine

The BSBRID subroutine is required if optimization is to be performed. Its function is to provide information required in the calculation of the sensitivities (gradients) of the bar and beam properties with respect to their dimensions.

Two basic types of information are provided. The first is the SENTYP array, which indicates how each section property varies as a function of each dimension. Values in the SENTYP array can be either: 0 for no variation; 1 for a linear variation; 2 for a nonlinear variation; 3 for an unknown variation. The SENTYP=3 option is to be used when you know that the property is a function of the design dimension, but analytical gradient information is not being provided using the BSBRGD subroutine. In this case, MSC Nastran will calculate the gradients for you using central differencing techniques.

The second piece of information is the ALIN array. This array provides any linear sensitivity data. For example, the C1 stress recovery location for the TUBE section is $1.0 * \text{DIM1}$ so that this sensitivity of this stress recovery point with respect to the first dimension is 1.0.

You may use your own naming convention for the subroutines that specify the section sensitivity data. The calling tree used for the MSC Nastran standard library is shown in [Figure B-3](#).

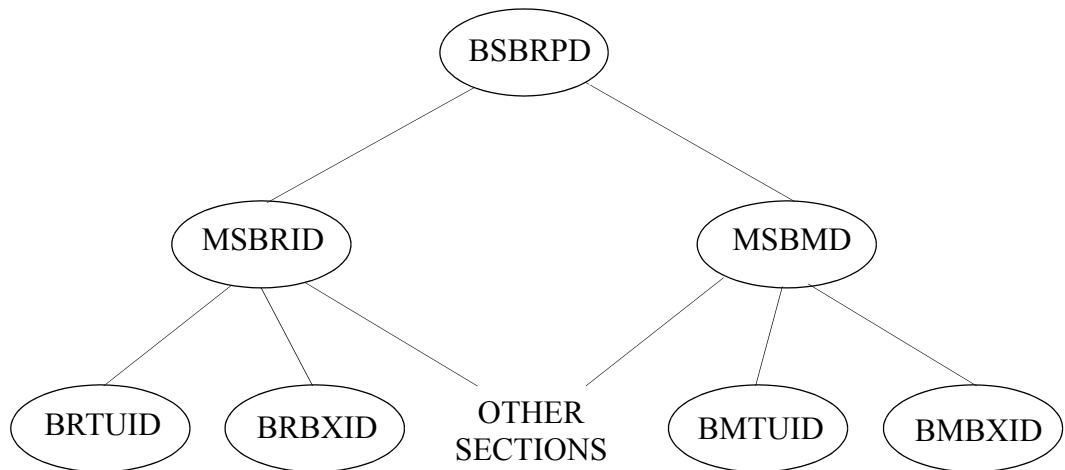


Figure B-3 Calling Tree to Generate Sensitivity Data

BSBRID calls the bar evaluator routine MSBRID/S for the BAR element and MSBMD/S for the BEAM element. The evaluators in turn call the routines for each section. The routines are named BRxxID or BMxxID, where xx is a two-letter identifier for the section. For example, the routine for the bar TUBE section is called BRTUID while the beam TUBE is called BMTUID. Listings of BSBRID, MSBRID, BRTUID and BMTUID are given in [Listing B-8](#), [Listing B-9](#), [Listing B-10](#), and [Listing B-11](#), respectively. While MSBMD is not shown, it can be developed by calling the BMXXID subroutines in place of the BRXXID of MSBRID.

Listing B-8 BSBRID Subroutine

```

SUBROUTINE BSBRID(GRPID,ENTYP,SECTON,SENTYP,ALIN,NDIM,NPROP,
1                   ERROR)
C =====
C Purpose
C   set up section dependent information for a particular cross
C   section type
C
C Arguments
C
C   GRPID  input integer      ID of the group
C   ENTYP  input integer      1: PBARL, 2: PBEAML
C   SECTON input character*8  Section type
C   SENTYP output integer    Type of sensitivity, 0: invariant,
C                           1: linear, 2: nonlinear
C   ALIN   output double     Matrix providing the linear
C                           factors for sensitive relationships
C   NDIM   input integer     Number of dimensions
C   NPROP  input integer     Number of properties
C   ERROR  output integer    Type of error
C
C Method
C       Simply transfer control based on entry type
C
C Called by
C       BCBRID
C
C Calls
C       MSBRID, MSBMID
C -----
IMPLICIT INTEGER (I-N)
IMPLICIT DOUBLE PRECISION (A-H,O-Z)

C Calling sequence arguments
INTEGER          ENTYP,ERROR,GRPID,SENTYP(NPROP,NDIM)
CHARACTER*8      SECTON
DOUBLE PRECISION ALIN(NPROP,NDIM)

C Local variables
INTEGER NAM(2)

C Local data
DATA NAM/4HBSBR,4HID  /
C =====
GRPID = 1
IF(ENTYP.EQ.1) THEN
  CALL MSBRID(SECTON,SENTYP,ALIN,NDIM,NPROP,ERROR)
ELSE if ( entyp .eq. 2 ) then
  CALL MSBMID(SECTON,SENTYP,ALIN,NDIM,NPROP,ERROR)
else
  ERROR = 5400
END IF

C
RETURN
END

```

MSBRID Subroutine

MSBRID is a brancher routine for providing information on the calculation of sensitivities for each of the bar types. You may rename this routine as you like or move its function to BSBRID. MSBRID calls the BRXXID routines, where XX is the two-letter keyword for various section types.



Listing B-9 MSBRID Subroutine

```

SUBROUTINE MSBRID(SECTON,SENTYP,ALIN,NDIM,NPROP,ERROR)
C =====
C Purpose
C   To set up section dependent information for PBARL cross section
C
C Arguments
C
C   SECTON input character*8 Name of section type
C   SENTYP output integer Type of sensitivity, 0: invariant,
C                           1: linear, 2: nonlinear
C   ALIN   output double Matrix providing the linear factors
C          for sensitive relationships
C   NDIM   input integer No. of dimensions
C   NPROP  input integer No. of properties in EPT data block
C   ERROR  output integer Type of error
C
C Method
C   Simply transfer the section dependent information of
C   the 19 kinds
C Called by
C   BSBRID
C
C Calls
C       BRRRID, BRBRID, BRTUID, BRBXID, BRIIID, BRLLID, BRTTID,
C       BRCHID, BRCRID, BRHHID, BRTIID, BRIIID, BRCIID, BRZZID,
C       BRC2ID, BRT2ID, BRB1ID, BRHXID, BRHTID, BRCLID
C       ZEROI, ZEROD (Nastran utility)
C -----
C
C IMPLICIT INTEGER (I-N)
C IMPLICIT DOUBLE PRECISION (A-H,O-Z)

C Calling sequence arguments
CHARACTER*8      SECTON
INTEGER          ERROR,SENTYP(NPROP,NDIM)
DOUBLE PRECISION ALIN(NPROP,NDIM)

C Local variables
INTEGER NAM(2)

C Local data
DATA NAM/4HMSBR,4HID  /
C =====
C
CALL ZEROI( SENTYP(1,1), NPROP*NDIM )
CALL ZEROD( ALIN(1,1), NPROP*NDIM )
ERROR = 0

C
IF(SECTON.EQ.'ROD') THEN
  CALL BRRRID(SENTYP,ALIN,NDIM,NPROP,ERROR)
ELSE IF(SECTON.EQ.'BAR') THEN
  CALL BRBRID(SENTYP,ALIN,NDIM,NPROP,ERROR)
ELSE IF(SECTON.EQ.'TUBE') THEN
  CALL BRTUID(SENTYP,ALIN,NDIM,NPROP,ERROR)
ELSE IF(SECTON.EQ.'BOX') THEN
  CALL BRBXID(SENTYP,ALIN,NDIM,NPROP,ERROR)
ELSE IF(SECTON.EQ.'I') THEN
  CALL BRIIID(SENTYP,ALIN,NDIM,NPROP,ERROR)
c10939 ELSE IF(SECTON.EQ.'L') THEN
c10939  CALL BRLLID(SENTYP,ALIN,NDIM,NPROP,ERROR)
ELSE IF(SECTON.EQ.'T') THEN
  CALL BRTTID(SENTYP,ALIN,NDIM,NPROP,ERROR)
ELSE IF(SECTON.EQ.'CHAN') THEN
  CALL BRCHID(SENTYP,ALIN,NDIM,NPROP,ERROR)

```



```

ELSE IF(SECTON.EQ.'CROSS') THEN
    CALL BRCRID(SENTYP,ALIN,NDIM,NPROP,ERROR)
ELSE IF(SECTON.EQ.'H') THEN
    CALL BRHHID(SENTYP,ALIN,NDIM,NPROP,ERROR)
ELSE IF(SECTON.EQ.'T1') THEN
    CALL BRT1ID(SENTYP,ALIN,NDIM,NPROP,ERROR)
ELSE IF(SECTON.EQ.'11') THEN
    CALL BRI1ID(SENTYP,ALIN,NDIM,NPROP,ERROR)
ELSE IF(SECTON.EQ.'CHAN1') THEN
    CALL BRC1ID(SENTYP,ALIN,NDIM,NPROP,ERROR)
ELSE IF(SECTON.EQ.'Z') THEN
    CALL BRZZID(SENTYP,ALIN,NDIM,NPROP,ERROR)
ELSE IF(SECTON.EQ.'CHAN2') THEN
    CALL BRC2ID(SENTYP,ALIN,NDIM,NPROP,ERROR)
ELSE IF(SECTON.EQ.'T2') THEN
    CALL BRT2ID(SENTYP,ALIN,NDIM,NPROP,ERROR)
ELSE IF(SECTON.EQ.'BOX1') THEN
    CALL BRB1ID(SENTYP,ALIN,NDIM,NPROP,ERROR)
ELSE IF(SECTON.EQ.'HEXA') THEN
    CALL BRHXID(SENTYP,ALIN,NDIM,NPROP,ERROR)
ELSE IF(SECTON.EQ.'HAT') THEN
    CALL BRHTID(SENTYP,ALIN,NDIM,NPROP,ERROR)
ELSE IF(SECTON.EQ.'HAT1') THEN
    CALL BRCLID(SENTYP,ALIN,NDIM,NPROP,ERROR)
ELSE
    ERROR = 5410
END IF

C
RETURN
END

```

BRTUID Subroutine

BRTUID is an example routine that shows how to define the sensitivity type of each of the bar properties for the tube and the subset of the sensitivities that are linear.

Listing B-10 BRTUID Subroutine

```

SUBROUTINE BRTUID(SENTYP,ALIN,NDIM,NPROP,ERROR)
C =====
C Purpose
C   To set up section dependent information for rod cross section
C
C Arguments
C
C   SENTYP output integer Type of sensitivity, 0: invariant,
C   1: linear, 2: nonlinear
C   ALIN   output double  Matrix providing the linear factors for
C                   sensitive relationships
C   NDIM   input  integer No. of dimensions
C   NPROP  input  integer No. of properties in EPT data block
C   ERROR  output integer Type of error
C
C Method
C       Simply provides the information
C
C Called by
C       MSBRID
C -----
IMPLICIT INTEGER (I-N)
IMPLICIT DOUBLE PRECISION (A-H,O-Z)

C Calling sequence arguments
INTEGER           ERROR,SENTYP(NPROP,NDIM)
DOUBLE PRECISION ALIN(NPROP,NDIM)

```



```

C      Local variables
      INTEGER NAM(2)

C      Local data
      DATA NAM/4HBRTU,4HID  /
C =====
C
C      ALIN( 7,1) =  1.0DO
C      ALIN(10,1) =  1.0DO
C      ALIN(11,1) = -1.0DO
C      ALIN(14,1) = -1.0DO
C      ALIN( 5,3) =  1.0DO
C
C      SENTYP( 1,1) = 2
C      SENTYP( 1,2) = 2
C      SENTYP( 2,1) = 2
C      SENTYP( 2,2) = 2
C      SENTYP( 3,1) = 2
C      SENTYP( 3,2) = 2
C      SENTYP( 4,1) = 2
C      SENTYP( 4,2) = 2
C      SENTYP( 7,1) = 1
C      SENTYP(10,1) = 1
C      SENTYP(11,1) = 1
C      SENTYP(14,1) = 1
C      SENTYP( 5,3) = 1

C
C      RETURN
C      END

```

BMTUID Subroutine

BMTUID is an example routine that shows how to define the sensitivity type of each of the beam properties for the tube and the subset of the sensitivities that are linear.

Listing B-11 BMTUID Subroutine

```

SUBROUTINE BMTUID(SENTYP,ALIN,NDIM,NPROP,ERROR)
C =====
C      Purpose
C      To set up section dependent information for TUBE cross section
C
C      Arguments
C
C      SENTYP output integer Type of sensitivity, 0: invariant,
C                               1: linear, 2: nonlinear
C      ALIN   output double  Matrix providing the linear factors for
C                           sensitive relationships
C      NDIM   input  integer No. of dimensions
C      NPROP  input  integer No. of properties in EPT data block
C      ERROR  output integer Type of error
C
C      Method
C              Simply provides the information
C
C      Called by
C              MSBMID
C -----
C      IMPLICIT INTEGER (I-N)
C      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
C
C      Calling sequence arguments
C      INTEGER           ERROR,SENTYP(NPROP,NDIM)

```



```
DOUBLE PRECISION ALIN(NPROP,NDIM)

C      Local variables
C      INTEGER NAM(2)

C      Local data
C      DATA NAM/4HBMTU,4HID   /
C =====
C      Tube dimensions for the beam:
C      1 - s0
C      2 - x/xb
C      3 - outer radius
C      4 - inner radius
C      5 - nsm

      ALIN(10,3) = 1.0D0
      ALIN(13,3) = 1.0D0
      ALIN(14,3) = -1.0D0
      ALIN(17,3) = -1.0D0
      ALIN( 9,5) = 1.0D0

C      SENTYP( 4,3) = 2
      SENTYP( 4,4) = 2
      SENTYP( 5,3) = 2
      SENTYP( 5,4) = 2
      SENTYP( 6,3) = 2
      SENTYP( 6,4) = 2
      SENTYP( 8,3) = 2
      SENTYP( 8,4) = 2
      SENTYP(10,3) = 1
      SENTYP(13,3) = 1
      SENTYP(14,3) = 1
      SENTYP(17,3) = 1
      SENTYP( 9,5) = 1

C      100  continue
C      call mprtd ( 'alin', alin, nprop,ndim )
C -----
      RETURN
      END
```



BSBRGD Subroutine

The BSBRGD subroutine is required if optimization is to be performed and analytical sensitivities are needed (SENTYPE = 2 in subroutine BSBRID). Its function is to provide the nonlinear gradients of the bar properties with respect to the bar dimensions. You may use your own naming convention for the subroutines that calculate the section gradients from the dimensions. The calling tree used for the MSC Nastran standard library is shown in [Figure B-4](#).

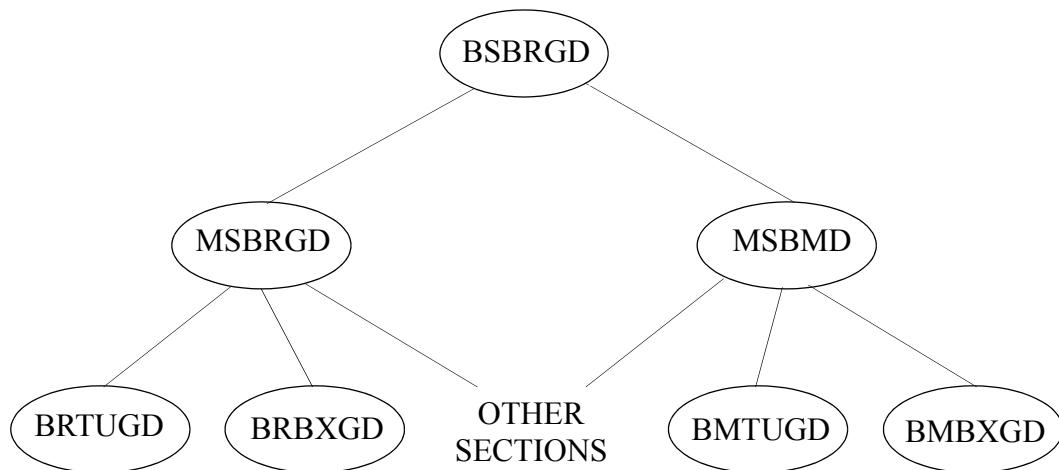


Figure B-4 Calling Tree to Generate Nonlinear Sensitivity Data

BSBRGD calls the bar evaluator routine MSBRGD/S for the BAR element and MSBMGD/S for the BEAM element. The evaluator, in turn, calls the routines for each section. The routines are named BRxxGD or BMxxGD where xx is a two-letter identifier for the section. For example, the routine for the TUBE section is called BRTUGD. Listings of BSBRGD, MSBRGD, BRTUGD and BMTUGD are given in [Listing B-12](#), [Listing B-13](#), [Listing B-14](#), and [Listing B-15](#), respectively. MSBMGD is not shown but can be developed by calling MBxxGD subroutines in place of the BRxxGD of MSBRGP.

Listing B-12 BSBRGD Subroutine

```

SUBROUTINE BSBRGD(GRPID,ENTYP,TYPE,DIMI,NDIMI,ANONL,NPROP,ERROR)
C =====
C Purpose
C   To get the nonlinear factors of sensitivities for default sections
C Arguments
C
C   GRPID  input  integer      ID of group name
C   ENTYP  input  integer      1: PBARL, 2: PBEAML
C   TYPE   input  character*8   Type name of cross-section
C   DIMI   input  double       Array from EPT record
C   NDIMI  input  integer      Number of dimensions (Plus NSM)
C   ANONL  output double      Array providing the nonlinear factors
C                           for sensitivity relationships
C   NPROP   input  integer      Number of properties in PBAR entries
C   ERROR  output integer      Type of error
C
  
```

```

C      Method
C          Simply transfer control based on entry types
C      Called by      BCBRGD
C
C      Calls          MSBRGD
C -----
C      IMPLICIT INTEGER (I-N)
C      IMPLICIT DOUBLE PRECISION (A-H,O-Z)

C      Calling sequence arguments
C      INTEGER           GRPID,ENTYP,NDIMI,NPROP,ERROR
C      CHARACTER*8        TYPE
C      DOUBLE PRECISION  ANONL(NPROP,NDIMI), DIMI(NDIMI)

C      Local variables
C      INTEGER NAM(2)

C      Local data
C      DATA NAM/4HBSBR,4HGD  /
C =====

C      GRPID = 1
C      IF(ENTYP .EQ. 1) THEN
C          CALL MSBRGD(TYPE,DIMI,NDIMI,ANONL,NPROP,ERROR)
C      ELSE if ( entyp .eq. 2 ) then
C          CALL MSBMGD(TYPE,DIMI,NDIMI,ANONL,NPROP,ERROR)
C      else
C          ERROR = 5300
C      END IF
C -----
C      RETURN
C

```

MSBRGD Subroutine

MSBRGD is a brancher routine for providing information on the calculation of nonlinear gradients for each of the bar types. You may rename this routine as you like or move its function to BSBRGD. MSBRGD calls the BRxxGD routines, where xx is the two-letter keyword for various section types.

Listing B-13 MSBRGD Subroutine

```

SUBROUTINE MSBRGD (TYPE,DIMI,NDIMI,ANONL,NPROP,ERROR)
C
C
=====
C      Purpose
C          To get the nonlinear factors of sensitivities for PBARL entries
C
C      Arguments
C
C          TYPE   input  character*8  Type name of cross-section
C          DIMI   input  double       Array from EPT record
C          NDIMI  input  integer     Number of dimensions (Plus NSM)
C          ANONL  output double    Array providing the nonlinear factors
C                                for sensitivity relationships
C          NPROP  input  integer     Number of properties in PBAR entries
C          ERROR  output integer    Type of error
C

```



```

C Method
C           Simply transfer information based on cross-section type
C Called by
C           BSBRGD
C
C Calls
C           BRRDGD, BRBRGD, BRBXGD, BRTUGD, BRIIGD, BRTTGD, BRLLGD, BRCHGD
C           BRCRGD, BRHHGD, BRT1GD, BRT2GD, BRI1GD, BRC1GD, BRC2GD, BRZZGD
C           BRHXGD, BRB1GD, BRHTGD, ZEROD (Nastran utility)
C -----
C -----
C           IMPLICIT INTEGER (I-N)
C           IMPLICIT DOUBLE PRECISION (A-H,O-Z)

C Calling sequence arguments
C           INTEGER          NDIMI, NPROP, ERROR
C           CHARACTER*8        TYPE
C           DOUBLE PRECISION ANONL(NPROP,NDIMI), DIMI(NDIMI)

C Local variables
C           INTEGER NAM(2)

C Local data
C           DATA NAM/4HMSBR, 4HGD   /
C
C =====
C =
C
C           CALL ZEROD( ANONL(1,1), NPROP*NDIMI )
C           ERROR = 0
C
C           IF(TYPE.EQ.'ROD') THEN
C               CALL BRRDGD(DIMI,NDIMI,ANONL,NPROP,ERROR)
C           ELSE IF(TYPE.EQ.'BAR') THEN
C               CALL BRBRGD(DIMI,NDIMI,ANONL,NPROP,ERROR)
C           ELSE IF(TYPE.EQ.'BOX') THEN
C               CALL BRBXGD(DIMI,NDIMI,ANONL,NPROP,ERROR)
C           ELSE IF(TYPE.EQ.'TUBE') THEN
C               CALL BRTUGD(DIMI,NDIMI,ANONL,NPROP,ERROR)
C           ELSE IF(TYPE.EQ.'I') THEN
C               CALL BRIIGD(DIMI,NDIMI,ANONL,NPROP,ERROR)
c10939 ELSE IF(TYPE.EQ.'L') THEN
c10939     CALL BRLLGD(DIMI,NDIMI,ANONL,NPROP,ERROR)
C           ELSE IF(TYPE.EQ.'T') THEN
C               CALL BRTTGD(DIMI,NDIMI,ANONL,NPROP,ERROR)
C           ELSE IF(TYPE.EQ.'CHAN') THEN
C               CALL BRCHGD(DIMI,NDIMI,ANONL,NPROP,ERROR)
C           ELSE IF(TYPE.EQ.'CROSS') THEN
C               CALL BRCRGD(DIMI,NDIMI,ANONL,NPROP,ERROR)
C           ELSE IF(TYPE.EQ.'H') THEN
C               CALL BRHHGD(DIMI,NDIMI,ANONL,NPROP,ERROR)
C           ELSE IF(TYPE.EQ.'T1') THEN
C               CALL BRT1GD(DIMI,NDIMI,ANONL,NPROP,ERROR)
C           ELSE IF(TYPE.EQ.'I1') THEN

```



```

        CALL BRI1GD(DIMI,NDIMI,ANONL,NPROP,ERROR)
ELSE IF(TYPE.EQ.'CHAN1') THEN
        CALL BRC1GD(DIMI,NDIMI,ANONL,NPROP,ERROR)
ELSE IF(TYPE.EQ.'Z') THEN
        CALL BRZZGD(DIMI,NDIMI,ANONL,NPROP,ERROR)
ELSE IF(TYPE.EQ.'CHAN2') THEN
        CALL BRC2GD(DIMI,NDIMI,ANONL,NPROP,ERROR)
ELSE IF(TYPE.EQ.'T2') THEN
        CALL BRT2GD(DIMI,NDIMI,ANONL,NPROP,ERROR)
ELSE IF(TYPE.EQ.'BOX1') THEN
        CALL BRE1GD(DIMI,NDIMI,ANONL,NPROP,ERROR)
ELSE IF(TYPE.EQ.'HEXA') THEN
        CALL BRHXGD(DIMI,NDIMI,ANONL,NPROP,ERROR)
ELSE IF(TYPE.EQ.'HAT') THEN
        CALL BRHTGD(DIMI,NDIMI,ANONL,NPROP,ERROR)
ELSE IF(TYPE.EQ.'HAT1') THEN
        CALL BRCLGD(DIMI,NDIMI,ANONL,NPROP,ERROR)
ELSE
        ERROR = 5310
ENDIF

C
RETURN
END

```

BRTUGD Subroutine

BRTUGD is a BAR example routine that shows how to define the nonlinear gradients of the TUBE section as a function of the outer and inner radii of the tube.

Listing B-14 BRTUGD Subroutine

```

SUBROUTINE BRTUGD(DIMI,NDIMI,ANONL,NPROP,ERROR)
C =====
C Purpose
C      To get the nonlinear factors of sensitivities for TUBE section
C
C Arguments
C
C      DIMI    input  double      Array of EPT records (Dimi+NSM)
C      NDIMI   input  integer     Number of dimensions (Plus NSM)
C      ANONL   output double    Array providing the nonlinear factors
C                                for sensitivity relationships
C      NPROP   input  integer     Number of properties in PBAR entries
C      ERROR   output integer    Type of error
C
C Method
C      Simply calculates the nonlinear factors
C Called by
C      MSBRGD
C -----
IMPLICIT INTEGER (I-N)
IMPLICIT DOUBLE PRECISION (A-H,O-Z)

C      Calling sequence arguments
INTEGER          NDIMI,NPROP,ERROR
DOUBLE PRECISION ANONL(NPROP,NDIMI), DIMI(NDIMI)

C      Local variables
INTEGER NAM(2)

```



```

C      Nastran common blocks
COMMON /CONDAD/ PI

C      Local data
DATA NAM/4HBRTU,4HGD  /
C =====
      DIM1      = DIM1(1)
      DIM2      = DIM1(2)
      PDIM13    = PI*DIM1**3
      PDIM23    = PI*DIM2**3
C
      ANONL(1,1) = 2*PI*DIM1
      ANONL(1,2) = -2*PI*DIM2
      ANONL(2,1) = PDIM13
      ANONL(2,2) = -PDIM13
      ANONL(3,1) = PDIM13
      ANONL(3,2) = -PDIM23
      ANONL(4,1) = 2*PDIM13
      ANONL(4,2) = -2*PDIM23
C
      RETURN
      END

```

BMTUGD Subroutine

BMTUGD is a BEAM example routine that shows how to define the nonlinear gradients of the beam TUBE section as a functional inner and outer radii of the tube.

Listing B-15 BMTUGD Subroutine

```

SUBROUTINE BMTUGD(DIMI,NDIMI,ANONL,NPROP,ERROR)
C
C =====
C      Purpose
C          To get the nonlinear factors of sensitivities for TUBE section
C
C      Arguments
C
C          DIMI   input  double      Array of EPT records (Dimi+NSM)
C          NDIMI  input  integer     Number of dimensions (Plus NSM)
C          ANONL  output double     Array providing the nonlinear factors
C                                for sensitivity relationships
C          NPROP  input  integer     Number of properties in PBEAM entries
C          ERROR  output integer    Type of error
C
C      Method
C          Simply calculates the nonlinear factors
C      Called by
C          MSBMGD
C -----
      IMPLICIT INTEGER (I-N)
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)

C      Calling sequence arguments
      INTEGER        NDIMI,NPROP,ERROR
      DOUBLE PRECISION ANONL(NPROP,NDIMI), DIMI(11 * NDIMI)

C      Local variables
      INTEGER NAM(2)

C      Nastran common blocks
COMMON /CONDAD/ PI

```



```

C      Local data
      DATA NAM/4HBMRD,4HGD  /
C =====
C      Tube dimensions for the beam:
C      1 - s0
C      2 - x/xb
C      3 - outer radius
C      4 - inner radius
C      5 - nsm
C      items 1 through 5 repeat for 10 additional stations

      ndims = 5
      do 20 isect = 0, 10
         irows = isect * 16 + 1
         idim  =  isect * ndims
         idim3 = idim + 3
         idim4 = idim + 4
C      special handling puts end b data
         if (  isept .lt. 10 .and. dimi(idim3 + ndims) .eq. 0 ) then
            irows = 161
         endif
C
         DIM1  = DIMI(idim3)
         DIM2  = DIMI(idim4)
         PDIM13 = PI*DIM1**3
         PDIM23 = PI*DIM2**3
C
         ANONL(irows+ 3,3) = 2*PI*DIM1
         ANONL(irows+ 3,4) = -2*PI*DIM2
         ANONL(irows+ 4,3) = PDIM13
         ANONL(irows+ 4,4) = -PDIM23
         ANONL(irows+ 5,3) = PDIM13
         ANONL(irows+ 5,4) = -PDIM23
         ANONL(irows+ 7,3) = 2*PDIM13
         ANONL(irows+ 7,4) = -2*PDIM23

C
         if ( irows .eq. 161) go to 30
20      continue
30      continue
C      Call mprtd ( 'nonlin',anonl,nprop,ndimi )

C
      RETURN
      END

```



BSBRCD Subroutine

The BSBRCD subroutine allows you to place constraints on values the beam dimensions can take during a design task. It is not needed unless optimization is used and, even then, is available only to impose conditions on the dimensions to keep the optimization process from selecting physically meaningless dimensions.

For example, the optimizer might select a TUBE design with the inner radius greater than the outer radius because this allows for a negative are and therefore a negative weight (something a weight minimization algorithm loves!). These constraints are not the same as the PMIN and PMAX property limits that are imposed on the DVPREL1 entry. Instead, these are constraints that occur between or among section dimensions. A DRESP2 entry could be used to develop the same design constraints, but the subroutine reduces the burden on the user interface, the primary goal of the beam library project. The calling tree used for the MSC Nastran standard library is shown in [Figure B-5](#).

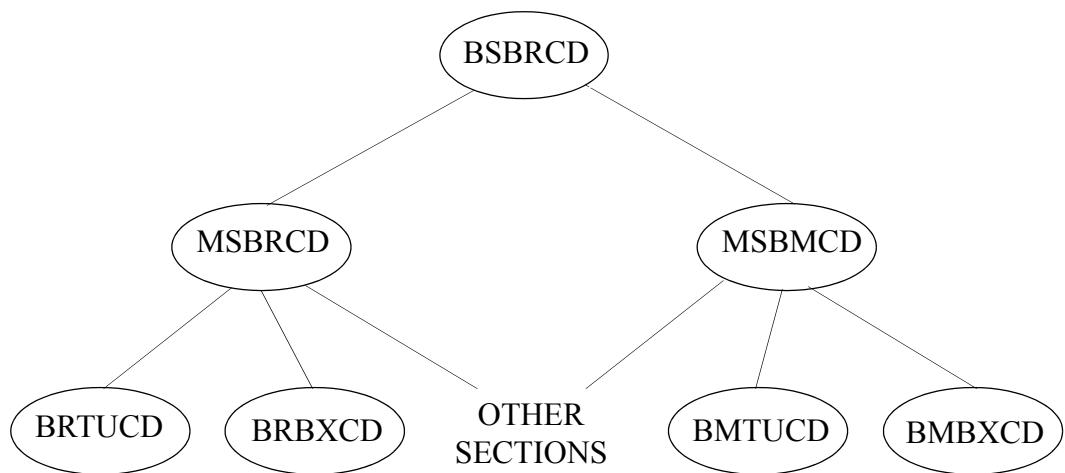


Figure B-5 Calling Tree to Generate Beam Dimension Constraints

BSBRCD calls the bar evaluator routine MSBRCD or the beam evaluator routine MSBMCD. The evaluator in turn calls the routines for each section that requires constraints. The routines are named BRxxCD or BMxxCD, where xx is a two-letter identifier for the section. For example, the routine for the TUBE section is called BRTUCD. Listings for BSBRCD, MSBRCD, BRTUCD and BMTUCD are given in [Listing B-16](#), [Listing B-17](#), [Listing B-18](#), and [Listing B-19](#), respectively. MSBMCD is not shown, but can be developed by calling the BMxxCD subroutines in place of BRxxCD of MSBRCD.



Listing B-16 BSBRC Subroutine

```

SUBROUTINE BSBRC (GRPID,ENTYP,TYPE,AFACT,NCONST,NDIMI,ERROR)
C
C =====
C Purpose          To get constraint information for default types
C
C Arguments
C
C     GRPID    input integer      ID of the group name
C     ENTYP    input integer      1: PBRL, 2: PBEAML
C     TYPE     input character*8   Section type
C     NCONST   input integer      Number of constraints
C                           for the section type
C     NDIMI   input integer      Number of dimensions
C                           for the section type
C     AFACT    output double     The factor for the NDIMI dimension in
C                           the constraint relation. Dimensions are
C                           NCONST by NDIMI.
C     ERROR   output integer     type of error
C
C Method           Simply transfers control based on PBRL or PBEAML entries
C
C Called by        BCBRCD
C
C Calls            MSBRC
C -----
IMPLICIT INTEGER (I-N)
IMPLICIT DOUBLE PRECISION (A-H,O-Z)

C     Calling sequence arguments
CHARACTER*8           TYPE
INTEGER                GRPID,ENTYP,NCONST,NDIMI,ERROR
DOUBLE PRECISION       AFACT(NCONST,NDIMI)

C     Local Variables
INTEGER NAM(2)

C     Local data
DATA NAM/4HBSBR,4HCD  /
C =====
C
GRPID = 1
IF(ENTYP.eq.1) THEN
    CALL MSBRC(TYPE,AFACT,NCONST,NDIMI,ERROR)
else
    CALL MSBMCD(TYPE,AFACT,NCONST,NDIMI,ERROR)
END IF
C -----
RETURN
END

```

MSBRC Subroutine

MSBRC is a brancher routine for providing information on the calculation of gradients for each of the bar types. You may rename this routine as you like or move its function to BSBRC. MSBRC calls the BRxxCD routines, where xx is the two-letter keyword for various section types.



Listing B-17 MSBRCD Subroutine

```

SUBROUTINE MSBRCD (TYPE, AFACT, NCONST, NDIMI, ERROR)
C
C =====
C Purpose          To get constraint information for PBARL types
C
C Arguments
C
C      TYPE    input character*8 Section type
C      AFACT   output double     The factor for the NDIMI dimension in
C                               the constraint relation. Dimensions are
C                               NCONST by NDIMI.
C      NCONST  input integer   Number of constraints
C                               for the section type
C      NDIMI   input integer   Number of dimensions
C                               for the section type
C      ERROR   output integer  type of error
C
C Method           Simply transfers control based on group name
C
C Called by        BSBRCT
C
C Calls            BRTUCD, BRBXCD, BRIICD, BRTTCD, BRLLCD, BRCHCD, BRCRCD,
C                  BRHHCD, BRT1CD, BRI1CD, BRC1CD, BRZZCD, BRC2CD, BRT2CD,
C                  BRB1CD, BRHXCD, BRHTCD, BRCLCD,
C                  ZEROD (Nastran utility)
C -----
C      IMPLICIT INTEGER (I-N)
C      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
C
C      Calling sequence arguments
C      CHARACTER*8           TYPE
C      INTEGER                NCONST,NDIMI,ERROR
C      DOUBLE PRECISION       AFACT(NCONST,NDIMI)
C
C      Local Variables
C      INTEGER NAM(2)
C
C      Local data
C      DATA NAM/4HMSBR,4HCD  /
C =====
C
CALL ZEROD( AFACT(1,1), NCONST*NDIMI )
ERROR = 0

IF(TYPE.EQ.'TUBE') THEN
  CALL BRTUCD(AFACT,NCONST,NDIMI,ERROR)
ELSE IF(TYPE.EQ.'BOX') THEN
  CALL BRBXCD(AFACT,NCONST,NDIMI,ERROR)
ELSE IF(TYPE.EQ.'I') THEN
  CALL BRIICD(AFACT,NCONST,NDIMI,ERROR)
c10939 ELSE IF(TYPE.EQ.'L') THEN
c10939  CALL BRLLCD(AFACT,NCONST,NDIMI,ERROR)
  ELSE IF(TYPE.EQ.'T') THEN
    CALL BRTTCD(AFACT,NCONST,NDIMI,ERROR)
  ELSE IF(TYPE.EQ.'CHAN') THEN
    CALL BRCHCD(AFACT,NCONST,NDIMI,ERROR)
  ELSE IF(TYPE.EQ.'CROSS') THEN
    CALL BRCRCD(AFACT,NCONST,NDIMI,ERROR)
  ELSE IF(TYPE.EQ.'H') THEN
    CALL BRHHCD(AFACT,NCONST,NDIMI,ERROR)
  ELSE IF(TYPE.EQ.'T1') THEN
    CALL BRT1CD(AFACT,NCONST,NDIMI,ERROR)

```



```

        ELSE IF(TYPE.EQ.'I1') THEN
            CALL BRI1CD(AFACT,NCONST,NDIMI,ERROR)
        ELSE IF(TYPE.EQ.'CHAN1') THEN
            CALL BRC1CD(AFACT,NCONST,NDIMI,ERROR)
        ELSE IF(TYPE.EQ.'Z') THEN
            CALL BRZZCD(AFACT,NCONST,NDIMI,ERROR)
        ELSE IF(TYPE.EQ.'CHAN2') THEN
            CALL BRC2CD(AFACT,NCONST,NDIMI,ERROR)
        ELSE IF(TYPE.EQ.'T2') THEN
            CALL BRT2CD(AFACT,NCONST,NDIMI,ERROR)
        ELSE IF(TYPE.EQ.'BOX1') THEN
            CALL BRB1CD(AFACT,NCONST,NDIMI,ERROR)
        ELSE IF(TYPE.EQ.'HEXA') THEN
            CALL BRHXCD(AFACT,NCONST,NDIMI,ERROR)
        ELSE IF(TYPE.EQ.'HAT') THEN
            CALL BRHTCD(AFACT,NCONST,NDIMI,ERROR)
        ELSE IF(TYPE.EQ.'HAT1') THEN
            CALL BRCLCD(AFACT,NCONST,NDIMI,ERROR)
        ELSE
            ERROR = 5500
    END IF

C
    RETURN
END

```

BRTUCD Subroutine

BRTUCD is an example routine that shows how to define the constraints for a bar section. This example routine is for the TUBE section and imposes a single constraint that $-DIM1 + DIM2 < 0.0$, where DIM1 is the outer radius and DIM2 is the inner radius of the tube. The constraints should always be specified so that the specified linear combination of dimensions is less or equal to zero when the constraint is satisfied.

Listing B-18 BRTUCD Subroutine

```

SUBROUTINE BRTUCD(AFACT,NCONST,NDIMI,ERROR)
C =====
C Purpose
C           To get constraint information for TUBE type
C
C Arguments
C
C     AFACT   output double      The factor for the NDIMI dimension in
C                               the constraint relation. Dimensions are
C                               NCONST by NDIMI.
C     NCONST  input integer      Number of constraints
C                               for the section type
C     NDIMI   input integer      Number of dimensions
C                               for the section type
C     ERROR   output integer    type of error
C
C Method
C           Simply transfers constraint information
C
C Called by
C           MSBRCD
C -----
C           IMPLICIT INTEGER (I-N)
C           IMPLICIT DOUBLE PRECISION (A-H,O-Z)
C
C Calling sequence arguments
C     INTEGER          NCONST,NDIMI,ERROR
C     DOUBLE PRECISION AFACT(NCONST,NDIMI)

```



```

C      Local Variables
      INTEGER NAM(2)

C      Local data
      DATA NAM/4HBRTU,4HCD  /
C =====
C
C      IF(NCONST.NE.1) THEN
C          ERROR = 5502
C          RETURN
C      END IF
C
C      AFACT(1,1) = -1.0D0
C      AFACT(1,2) =  1.0D0

C
C      RETURN
C

```

BMTUCD Subroutine

BMTUCD is an example routine that shows how to define the constraints for a beam section. This example routine is for the TUBE section and imposes a single constraint that $-DIM1 + DIM2 < 0.0$, where DIM1 is the outer radius and DIM2 is the inner radius of the tube. The constraints should always be specified so that the specified linear combination of dimensions is less or equal to zero when the constraint is satisfied.

Listing B-19 BMTUCD Subroutine

```

SUBROUTINE BMTUCD(AFACT,NCONST,NDIMI,ERROR)
C
C =====
C      Purpose
C          To get constraint information for TUBE type
C
C      Arguments
C
C          AFACT    output double      The factor for the NDIMI dimension in
C                                      the constraint relation. Dimensions are
C                                      NCONST by NDIMI.
C          NCONST   input integer     Number of constraints
C                                      for the section type
C          NDIMI    input integer     Number of dimensions
C                                      for the section type
C          ERROR    output integer   type of error
C
C      Method
C          Simply transfers constraint information
C
C      Called by
C          MSBMCD
C -----
C          IMPLICIT INTEGER (I-N)
C          IMPLICIT DOUBLE PRECISION (A-H,O-Z)
C
C      Calling sequence arguments
C          INTEGER           NCONST,NDIMI,ERROR
C          DOUBLE PRECISION  AFACT(NCONST,NDIMI)
C
C      Local Variables
C          INTEGER NAM(2)
C
C      Local data

```



```
        DATA NAM/4HBMTU,4HCD  /
C =====
C      IF(NCONST.NE.1) THEN
C          ERROR = 5502
C          RETURN
C      END IF
C
C      AFACT(1,3) = -1.0D0
C      AFACT(1,4) =  1.0D0
C
C      RETURN
C      END
```



BSMSG Subroutine

The Error handling is performed by a subroutine called bsmsg.f. This routine has the following parameters.

```
SUBROUTINE BSMSG (GRPID,ERRCOD,MXLEN,Z,ERROR)
C =====
C Purpose
C   To handle the error message of 'MSCBML0' group
C -----
C Arguments
C
C   GRPID  input  int      ID of group or group name
C   ERRCOD input  int      Error message number if any found
C   MXLEN  output int     Maximum length of the message that can be passed
C   Z      output char    Array to contain the message return
C   ERROR  output int     The code returned indicates the type of error
C
C Reassigned
C
C Called by
C           BMSG routine
C-----
```

The purpose of this subroutine is to return an error String associated with an error code. The error codes are to be returned by the other 7 “BS...” routines and, as such, the “BSMSG” routine is used as the repository of all of the error messages for the Beam Library applications.

For example, suppose that “BSRPD” application returns an error code of 5103 when a certain error condition occurs. Then, the Beam Library Client routines will expect that there will be String returned from the “BSMSG” routine, which corresponds to this error code. These error messages will be printed in the “*.f06” file to guide the user as to what the error could have been and how to fix it. The string is limited to be no more than 139 characters.

The following is an example of BSMSG code construct:

```
IF (ERRCOD .EQ. 5104) THEN
    Z(1:MXLEN) = 'this is a User specified error message...//'
    &   ' Message can be up to 139 character long..'
    ERROR = ERRCOD
ELSE IF ...
    ...
END
```

We recommend that you use the example Beam Server files as a template to generate your BSMSG routines.



Linking Your Library to MSC Nastran

Once you have created the eight “BS...” routines, these routines may be linked with MSC Nastran Beam Server Library to build a beam server executable. An example user-defined beam server has been delivered with MSC Nastran. It is highly recommended that you study, build and use the example “Beam Server” before you build your own version of the beam server.

The MSC Nastran special library contains a main routine as well as the communications routines that allow MSC Nastran to communicate with the user-defined beam server.

The MSC Nastran user may connect up to 10 beam servers in a single job execution. This connection is made using the concept of evaluator groups described in the remainder of this section. For each group, the user specifies on the PBEAML/PBTRL entry referring to an external beam server, MSC Nastran will start and communicate with the beam server.

You may define as many beam evaluators as required using the “CONNECT FMS commands. Only 10 of these evaluators, however, may be referenced in the groups on the PBTRL or PBEAML Bulk Data entries.

The PBTRL/PBEAML entries specify a “GROUP” name on the fourth field of the first entry. This group name is associated with an “Evaluator” class using a “Connect” command in the FMS section. Finally, the “Evaluator” class is associated with an executable using the MSC Nastran configuration file specified via the “gmconn” keyword on the nastran command line.

The following example shows the mappings mentioned:

1. Group is referenced on the PBTRL/PBEAML entry (or entries).
`PBTRL,39,6,LOCserv,I_SECTION` (specify the “Group” name).
2. The “Group” is associated with an “Evaluator” class.
`CONNECT,BEAMEVAL,LOCserv,EXTBML` (Associate the “Group” name with an “Evaluator” class.)
3. The external evaluator connection file associates the “Evaluator” class with a server executable.
The following statement must be specified in the connection file:
`EXTBML,-,beam_server_pathname`
4. Refer to the external evaluator connection file on the command line using the “gmconn” keyword.
`nastran myjob...gmconn=external_evaluator_pathname`
5. In the example Beam Library section note that the standard input (FORTRAN unit 5) or standard output (FORTRAN unit 6) are not used as these I/O channels are reserved by the Inter-Process Communications (IPC) subsystem.



Example of Building and Linking a Beam Server

As an example of building and linking a beam server executable, the sample beam server will be modified. Complete instructions on building and using a beam server are provided in the *MSC Nastran Installation and Operations Guide* for each release of MSC Nastran.

- Make a copy of the beam server sample source.
- Edit the source for the BRTUPD subroutine; this routine describes the equations that convert the PBARL dimensions into the standard PBAR dimensions for a tube cross section.
- Add an extra multiplication of 3.0 to the DIMO(2) equation to increase the calculated moments of inertia.

Since the formulation of this bar section has been changed, the sensitivities for optimization will also change. Rather than calculate what the new sensitivities, MSC Nastran can calculate them using central differencing techniques. To permit this, edit the source file for the BRTUID subroutine and change all occurrences of SENTYP = 2 to SENTYP = 3.

Build your new beam server using the instructions detailed in the Installation and Operations Guide for your system. Once you have built the beam server executable, you must create an external evaluator connection to point to your executable. Typically, this file would be kept in the user's home directory, but for this example it will remain in the current directory. Edit the new file bmconfig.fil. Put the following line in the file:

`LOCBMLS, -, pathname`

where LOCBMLS is the evaluator referenced in the SAMPLE data file included with the beam server. Remember, this file can contain references to any number of beam servers.

To run the sample job, type in the following command:

```
nastran sample scr=yes bat=no gmconn=bmconfig.fil
```

Common problems which may occur when attempting to run an external beam library job are generally indicated in the F06 by USER FATAL MESSAGE 6498. If this message includes the text "No such group defined," the PBARL/PBEAML selected a group not defined on a CONNECT entry. If UFM 6498 includes the text "No such evaluator class," either the "gmconn" keyword was not specified or the CONNECT entry selected an evaluator not defined in the configuration file.

If the job was successful, you can look at the Design Variable History and see that the results for the variable mytubeor are different than the results for tubeor. These variables refer to the other radius of tube sections from equivalent models. One model used the MSC Nastran tube section while the other used the tube section in your modified beam server.





C

The IPOPT Algorithm

■ IPOPT 738



IPOPT

Introduction

MSC Software conducts surveys of optimizer technologies from industry and academia. This has led to the integration of optimizer IPOPT. IPOPT implements an interior point line search filter method that aims to find a local optimal solution for large scale nonlinear optimization. It was originally developed by Carnegie Mellon University in 2002 and now is supported by IBM.

Interior point method as one of barrier methods was first proposed in the sixties. Barriers methods are used to transform a “difficult” constrained problem into a sequence of “easy” unconstrained problems. MSCADS SUMT method is one of this classical barrier method. Barrier methods were popular during the sixties.

Earlier, the classical barrier method had its shortcomings; practitioners of nonlinear programming lost interest and switched to the more efficient MMFD and SQP-like methods (MSCADS and DOT) in the mid-seventies and eighties.

In the mid-eighties, a modern interior-point revolution started with the well-known Karmarkar linear programming algorithm which can be interpreted as a barrier method. Since then, interior point algorithms have emerged as one of most important and useful algorithms for mathematical programming. In particular, these interior point methods provide an attractive alternative to active constraint set methods in handling problems with large numbers of design variables and inequality constraints.

IPOPT is a software package for large scale nonlinear optimization. This code has been shown to be capable of handling tens of thousands of design variables. It can be used to solve SOL 200 sizing, shape, topology, topometry, and topography problems.

Benefits

The interior point method is a very robust algorithm that provides an alternative to SOL 200 existing optimizers, in particular, MSCADS SUMT method. The IPOPT optimizer not only enables performing practical topology, topometry, and topography optimization tasks but can also be used to perform standard shape and sizing optimization for design tasks.

Theory

In this section, a very brief discussion about the interior point method implemented in IPOPT is presented. More detail can be read in “*On the implementation of a primal-dual interior point filter line search algorithm for large-scale nonlinear programming*”, Mathematical Programming, 106(1):25–57, 2006 by A. Wachter and L. T. Biegler.

To simplify the description of the interior point method, we consider a problem with equality constraints (inequality constraints can be transformed to equality constraints by introducing slack variables) as:



$$\begin{aligned}
 & \text{minimize} && f(x) \\
 & \text{subject to} && C_j(X) = 0 \quad j = 1, 2, \dots, m \\
 & && X_i \geq 0 \quad i = 1, 2, \dots, n
 \end{aligned} \tag{C-1}$$

where X^L and X^U are the lower and upper bounds on the design variables X , n is the number of design variables, and m is the number of equality constraints. The objective function $f(X)$ and the equality constraints $C_j(X)$ are assumed to be twice continuously differentiable.

In general, gradient-based optimization algorithms have a common strategy as below:

A general optimization algorithm loop

- Start $k = 0$, $X = X^0$
- Evaluate $f(X)$ and $C_j(X)$
- Calculate gradients of $f(X)$ and $g_j(X)$
- Determine a search direction d_k
- Perform a one-dimensional search to find α^* that will minimize $f(X + d_k)$ subject to the constraints.
- Set $X^{k+1} = X^k + \alpha^* d_k$
- Check for convergence. If satisfied, exit. Otherwise repeat the loop

Two critical parts of the optimization task consists of determining a search direction and finding a best one-dimensional search step. The determining a search direction is the most time consuming part and one of major difference between the interior method and SOL 200 other optimization methods.

As a barrier method, the interior point algorithm computes (approximates) solutions for a sequence of barrier problems

$$\begin{aligned}
 & \text{minimize} && \varphi_\mu(x) = f(x) - \mu \sum_{i=1}^n \ln(x^i) \\
 & \text{subject to} && C_j(x) = 0 \quad j = 1, \dots, m
 \end{aligned} \tag{C-2}$$

for a decreasing sequence of barrier parameters μ converging to zero. Equivalently, this can be interpreted as applying a homotopy method to the primal-dual equations,

$$\begin{aligned}\nabla f(x) + \nabla c(x)\lambda - z &= 0 \\ c(x) &= 0 \\ XZe - \mu e &= 0\end{aligned}\tag{C-3}$$

with the homotopy parameter μ which is driven to zero. Here, $\lambda \in \mathbb{R}^m$ and $z \in \mathbb{R}^n$ correspond to the Lagrangian multipliers for the equality constraints and the bound constraints, respectively. Note, that Eq. (C-3) for $\mu = 0$ together with “ $x; z \geq 0$ ” are the Karush-Kuhn-Tucker (KKT) conditions for the original problem Eq. (C-2). Those are the first order optimality conditions for Eq. (C-1) if constraint qualifications are satisfied Eq. (C-3).

In order to solve the barrier problem Eq. (C-2) for a given fixed value μ_j of the barrier parameter, a damped Newton's method is applied to the primal-dual Eq. (C-3). Here, a search direction is obtained from solving a symmetric linear system

$$\begin{bmatrix} W_k + \Sigma_k + \delta_w I & A_k \\ A_k^T & -\delta_c I \end{bmatrix} \begin{bmatrix} d_k^x \\ d_k^\lambda \end{bmatrix} = \begin{bmatrix} \nabla \varphi_{\mu_j}(x_k) + A_k \lambda_k \\ c(x_k) \end{bmatrix} \tag{C-4}$$

where Jacobian $A_k = \nabla c(x_k)$ and W_k denotes the Hessian $\nabla_{xx}^2 L(x_k, \lambda_k, z_k)$ of the Lagrangian function $L(x, \lambda, z) = f(x) + c(x)^T \lambda - z$

The choice of scalars δ_w and δ_c is discussed in Wachter's paper.

The overall efficiency of the interior point method is dependent on solving a $(n+m) \times (n+m)$ sparse linear system Eq. (C-4).



Symbols
.asm file 626

Numerics
10 bar truss 647

A
ACHOBJ 101
ACINTS 230
Acoustic Optimization 559
ACPWR 229
Active and violated constraints 75
Active and violated constraints (CT, CTMIN) 244
Active constraints from type-1 responses 94
Active constraints on type-2 responses 95
Adjoint sensitivity analysis 66
Advanced numerical techniques in SOL 200 650

AEMOND1 232
AEMONP1 232
AESO 621
AFINTS 230
AFPRES 230
AFPWR 230
AFVELO 230
Alternative buckling response 579

Analysis
 sensitivity 30
 structural 29

Analysis discipline definition 137

Analysis model description
 cantilevered beam 514
 cantilevered plate 519, 598
 stiffened plate 528
 three-bar truss 491
 twenty-five bar truss 544

Analysis versus design Optimization 4

Analytic boundary shapes
 auxiliary models 331
 basis vectors in shape Optimization 314
 relating design variables to shape changes 322

shapes Optimal design 353
Application examples 395
Approximate model 30, 78
Approximate Optimization (DELP, DPMIN, DELX, DXMIN) 242
Approximation concepts used in structural Optimization 23
APRCOD 235
ARBITRARY BEAM CROSS SECTION 679
Associating FE with surface 411
Automatic external superelement Optimization (AESO) 621
Automatic updates of move limits 84
Auxiliary models in shape Optimization 316
B
Basic compliance minimization 397, 415
Basic Optimization problem statement
example 12
Basics
numerical Optimization 9
Basis vectors in shape Optimization 311
Bead or Stamp 468
Bead pattern 477
BEADVAR 143
Benefit of synthetic and external responses 45
Benefits
Topography Optimization 468
Topometry Optimization 457
trust region 611
BMTUCD 731
BMTUGD 725
BMTUID 719
BMTUPD 706
BMXXPD 703
BNDGRID 143
Boundary shape Changes using auxiliary boundary models 355
Bridge example 395
Bridge topology design 397
Brief literature review 373
BRTUCD 730
BRTUGD 724
BRTUID 718

Subroutine 718
BRTUPD 703
BRXXPD 703
BSBRCD 727
BSBRGB 721
BSBRID 715
BSBRPD 700
BSBRT 712
BSCON 698
BSGRQ 711
BSMSG 733
Buckling load factor sensitivities 69
Building an external response server 254
Bulk data and parameters 375
Bulk data entries 143
Bulk data entry DRESP1 378
C
Cantilevered plate 518
Car model topometry, example 464
Case control 618
Case control for design Optimization 111
Case control section 137
Casting constraints
 one die 422
 two dies 423
CBAR 35
CBEAM 35
CBUSH 35
CDAMP2,4 35
CEIG 221
CELAS2 35
CELAS4 35
CFAILURE 225
CGAP 35
CHGDV 101
CHGOBJ 101
CHGPRP 101
Choice of approximation method (APRCOD) 235

CMASS2,4 35
Comma separated values file 303
COMP 221
Complex eigenvalue sensitivity analysis 70
Compromise infeasible design 104
Concepts
 approximation used in structural 23
CONM1 35
CONM2 35
CONMAX 101
Connectivity properties
 table 35
CONROD 35
Constraint
 deletion 47
 normalization (GSCAL) 243
 regionalization 48
 screening 30
Constraint screening 47
Constraints
 beam dimensions 91
 beam library 92
 dependent design variables 91
 properties 91
Continuing the design process in a Subsequent Job 590
Converged 31
Convergence criteria (CONV1, CONV2, GMAX, CONVDV, CONVPR) 242
Convergence criteria parameters 101
Convergence of design cycles
 hard and soft convergence 98
Convex linearization 88
Coupled fluid-structure interaction sensitivity 64
Coupling analysis and Optimization using approximations 25
CQUAD4,8,R 35
Creating
 analysis deck 437
 BEADVAR 481
CONNECT command 260
constrained merit function 612

constraint set 435
constraint sets 505
constraints 503
coordinate frame 399
design study 436, 506
design variables 497
displacement constraint 433
DRESP3 entry 248
evaluator connection file 261
Isosurface 442
load case 400
mass constraint 432
new surface 410
normal modes constraint 504
objective 431, 502
objective and constraint (Patran) 390
objectives and constraints 402
subcases 439
topology variable 389
TOPVAR 429
variable linking (DLINK) 499
variable linking (property relation) 500
Creating new surface 410
Creation and selection of subcase 406
Creation of auxiliary model, and generation of basis vectors 344
CSTRAIN 224
CSTRAT 225
CSTRESS 225
CTRIA3,5,R 35
CTRIAX6 35
D
Data recovery 58
DCONADD 147
DCONSTR 148
DDVAL 151
Default values for DOPTPRM 381
Define property relation 501
Defining

constraints 129
design variables 113
objective function 127
DELP 83
DELXESL 640
Density distribution plot 394
DEQATN 152
Description
 SOL 200 input file 645
 SOL 400 input file 645
 template routines 263
Descriptions
 individual include files 646
Design and analysis properties (PTOL, PLVIOL) 242
Design constraint identification 138
Design constraints 40
Design cycle history
 continued design task 593
 DSOUG6 365
 DSOUG9 556
Design cycle print controls 239
Design domain 486
Design history
 DSOUG3 525
Design history output
 DSOUG4 532
Design history results
 DSOUG8 550
Design model 33
Design model description
 cantilevered plate 519, 599
 stiffened plate 531
 three-bar truss 491
 twenty-five bar truss 545
Design modeling
 overview 109
Design move limits in ESLNRO 636
Design objective 40
Design objective identification 138

Design Optimization
 input 347
 introduction 2
 output 269
Design Optimization parameters (DOPTPRM) - TCHECK and TDMIN 379
Design Optimization versus analysis 4
Design Optimization with composite materials 552
Design punch output 297
Design response characterization 141
Design responses 39
Design responses in superelementdesign modeling 618
Design sensitivity
 introduction 2
Design sensitivity analysis 50
Design sensitivity and Optimization examples
 complex spacecraft conceptual design stage 4
 connecting rod production 4
 frame structure overstressed 4
 frame structure sensitive instruments 4
Design sensitivity coefficient -- graphical interpretation 51
Design sensitivity output 291
Design sensitivity output example 294
Design sensitivity print 140
Design sensitivity response attributes
 table 215
Design sensitivity and Optimization
 introduction 2
Design task 364
Design task definition 138
Design variable limits 81
Design variable set selection 139
Design variables 33
Design variables and the finite element model 114
Design variables in superelement design modeling 617
Designating the design responses 119
Designed properties 33
 type-1 36
 type-2 38

type-3 38
Designed responses
 type-1 39
 type-3 40
Designed shapes 39
deslo_desmod.blk 646
deslo_desvar.blk 646
deslo_esl.blk 646
deslo_eslsub.cas 646
deslo_grid.blk 646
deslo_loads.blk 646
deslo_model 646
deslo_nlmat.blk 646
deslo_nlsub.ca 646
DESMAX 241
DESPCH 267, 298
DESPCH1 267
DESVAR 84, 155
Developments from version 68 iv
Direct and adjoint sensitivity methods 53
Direct approximations 87
Direct densitivities of static responses 55
Direct frequency response sensitivity analysis 61
Direct input of shapes 314, 320, 325
Discrete variable Optimization 606
Discrete variable Optimization (DISCOD, DISBEG) 243
DISP 222
Display of fringe plot 408
Displaying density distribution 393
DLINK 157
dmp 657
DOF domain parallel lanczos method 656
DOMAINSOLVER 657
DOPTPRM 83, 159, 235
DOPTPRM design Optimization parameters 380
DRESP1 159, 215
DRESP1 bulk data entries 119
DRESP2 165
DRESP2 bulk data entries 119

DRESP3 170
DRESP3 bulk data entries 125
DSA 657
DSCREEN 174
DSMXESL 640
DSOUG1C 594
DSOUG1R 596
DSOUG4 output 269
DTABLE 176
DTABLE Bulk Data entry
specification of 177
DTABLE2 177
DVBSHAP 178
DVCREL1 179
DVCREL2 182
DVGRID 184
DVMREL1 187
DVMREL2 190
DVPREL1 192
DVPREL2 194
DVPSURF 197
DVSHAP 197
Dynamic response Optimization 533
E
Efficient definition of responses 121
EIGN 221
Element and grid responses 53
Element properties
table 34
Engine mount 425
Equality constraints in MSC Nastran 42
ESE 223
ESLNRO
design move limits 636
implementation 637
ESLOPT (443) 639
Evaluator connection file extconnect example 261
Example

basic Optimization problem statement 12
building and inking a beam server 735
car model tomometry 464
design sensitivity output 294
DESPCH 298
shape basis vectors 323
shape Optimization 343
Square (toge1.dat) 473
structural 17
Three-bar truss 460
Toptimization 478
toptimization 483
trust region 615
type-1 properties 36
Executive control 136
Exporting IGES file 413
External response (Type-3 Response) 245
External responses 135
F
FE with surface 411
Features and benefits
 topology Optimization 372
File management 134
Final sensitivity calculations 58
Finite difference methods 53
Finite difference perturbations (DELB, STPCSL) 242
FLUTTER 233
Flutter sensitivity analysis 71
FORCE 224
Formal approximations 78
Formatted design sensitivity 291
FRACCL 228
FRDISP 226
FREQ 222
Frequency domain parallel lanczos method 656
FRFORC 228
FRMASS 221
Front-Mount-Beam 449
Front-mount-beam 426

- Front-mount-beam FE model 426
- FRSPCF 228
- FRSTRE 228
- FRVELO 227
- FSDALP 609
- FSDMAX 609
- Fully stressed design 608
- Fully stressed design (FSDMAX, FSDALP) 241
- Function evaluation 86
- Fundamentals
 - overview 28
- G
- General considerations 50, 61
- General topology Optimization preprocessing 387
- Generation of a new bulk data file 142
- Geometric boundary shapes 314, 321, 329
- Glossary of terms 685
- gmconn keyword 261
- GPFORCE 225
- GPFORCP 226
- GPLY 34
- Gradient
 - evaluation 93
- Gradients
 - active beam library constraints 97
 - active dependent design variables 96
 - active property constraints 96
 - active type-3 responses 96
 - properties 93
 - type-1 responses 94
 - type-2 responses 95
- Guidelines and limitations 614
 - Topography Optimization 472
 - Topometry Optimization 460
- H
- Hard convergence decision logic 101
- I
- Illustration 9

CT and CTMIN 77
Implied forcing frequency argument 154
Importing model into patran 398
Improved design 30
Inertia relief sensitivity analysis 58
Initial design 29
Initiating toptimize 454
Initiating toptimize for Topology Optimization 401
Input
 shape basis vectors 313
 Topography Optimization 468
 Topometry Optimization 458
Input file
 assembly run 626
 DSOUG 491
 DSOUG10 561
 DSOUG11 567
 DSOUG12 575
 DSOUG13 580
 DSOUG2 514
 DSOUG3 520
 DSOUG7 535
 DSOUG8 546
 DSOUG9 554
 glued contact 452
 MBB beam 416
 torsion beam with extrusion 421
Insight preferences 440
Internal parallel for FRRD1 module 655
Internet resources x
Introduction
 adding your own beam cross section library 696
 design Optimization 2
 design sensitivity 2
 design sensitivity and Optimization 2
 MBB beam 414
 Topography Optimization 468
 Topology Optimization 372
 Topometry Optimization 457

- torsion beam 419
- trust region 611
- IPRINT 240, 267
- Isosurface 444
 - attributes 444
 - control 446
- L
- LAMA 222
- Limitations 386
 - Optimizer 7
 - superelement Optimization 620
- Linear design space 79
- Linking your library to MSC.Nastran 734
- List
 - MSC Nastran books viii
- M
- Managing the structural Optimization task 22
- Manual grid variation 313, 319
- Manufacturing constraint 390
- Manufacturing constraints 482
- MAT1 35
- MAT10 35
- MAT2 35
- MAT3 35
- MAT8 35
- MAT9 35
- Material properties
 - table 35
- Matrix domain automated component modal synthesis (MDACMS) 658
- Maximum number of mathematical programming design cycles (DESMAX) 241
- MBB beam topology design 416
- MBB beam with variations 414
- MDACMS 658
- METHOD options
 - table 237
- Methods 237
- MEVBMD 701
- MEVBRD 701

Minimum member size control, MBB 416
Mirror symmetric constraints 397
Mirror symmetric constraints, MBB 417
Modal frequency response analysis 62
Modal transient response sensitivities 63
Mode tracking 46, 140
Model 1 - Plate with hole 336
Modeling considerations 343
Modeling guidelines and limitations 384
Modeling methods (Shape basis vector definition) 318
Modeling summary 363
Modeling tips 384
Modification of move limit parameters 286
Modified DOPTPRM default for Topology Optimization
table 244
Modified objective function 633
MODTRAK 198
MONPNT3 232
Move limits 81, 84
MSBRCD 728
MSBRGD 722
MSBRID 716
MSC Nastran
 equality constraints 42
 sensitivity coefficients 51
MSC Nastran books viii
MSCADS options
 table 237
Multi Model Optimization 664
Multidisciplinary analysis 43
Multidisciplinary Optimization 43
N
NASPRT 267
Nastran database files 625
Nastran result files 644
New bulk data file 300
Nomenclature 682
Non-Unique optima 103
Normalized constraints 40

Numerical Optimization
 basics 9
Numerical searching 13, 75
Numerically searching for an optimum 13
O
 Objective and constraint value evaluation 90
 One dimensional search 13
 OpenMDO 679
 OPTCOD 133
 OPTCOD options design Optimization parameters 236
 OPTEXIT 267
 Optimization 2, 74, 397
 parameters 404
 structural 22
 Optimization method (OPTCOD, METHOD) 236
 Optimization of nonlinear structural responses 633
 Optimization problem statement 11
 Optimization results 364
 Optimization solution 397
 glued contact 451
 MBB beam 415
 Patran tutorial 2 427
 torsion beam with variations 421
 Optimized bead pattern 477
Optimizer 30
 Optimizer and the approximate model 74
 Optimizer and the approximatefunction and gradient evaluations 75
 Optimizer level print control (IPRINT) 240
 Optimizer limitations 7
 Optimizers (Licensing) 133
Output
 DSOUG1 495
 DSOUG2 516
 Topography Optimization 472
 Topology Optimization 381
 Topometry Optimization 460
Output control 267
Output from PARAM POST 304

Overview

fundamentals 28

P

P1 267

P2 267

P2CALL 267

P2CBL 267

P2CC 267

P2CDDV 267

P2CM 267

P2CP 267

P2CR 267

P2RSET 268

PACABS 34

PACBAR 34

Parallel Sensitivities 657

Parameters related to trust region in DOPTPRM entry 613

Parameters unique to design sensitivity and Optimization 212

Patran

post-process illustration 392

postprocessing 391

pre-process illustration 388

tutorial 2 427

user interface 387

PBAR 34, 660

PBAR record 703

PBART 34, 661

PBART record 703

PBEAM 34, 640, 662

PBEAM record 706

PBEAML 34, 663

PBEAML record 706

PBRSECT/PBMSECT 34

PBUSH 34

PBUSH1D 34

PCOMP 34

PCOMPG 34

PDAMP 34

PELAS 34

PENAL 238
Penalty weight 612
Perturbed and baseline solution vectors 57
PGAP 34
Pinned-pinned column buckling 579
PMASS 34
PMAX 83
PMIN 83
PMOVE 83
POST 268
Postprocessing output 303
Postprocessing topography results 475
PRES 227
Primary Nastran result files 644
Print controls 239
Printout from DOM12 613
PROD 34, 640
Properties
 type-1 36
 type-2 38
 type-3 38
Property and/or shape basis vector perturbation 55
Property evaluation 86
Property limits 82
Property Optimization design task 590
Property to Optimize 485
PSD response 44
PSD response sensitivities 72
PSDACCL 229
PSDDISP 228, 229
PSDVELO 229
Pseudo-Load vectors 56
PSHEAR 34
PSHELL 34, 640
PTUBE 34, 640
Punch parameters 297
PVISC 34

Q
QUAD4 element with linear output 223

R
R3SGRT 583
R3SVALD 251, 584
Randomization of a user's input data file 631
Reading density distribution 392
Reading density distribution file into Patran 407
Real eigenvalue sensitivities 68
Real eigenvector sensitivities 68
Reciprocal approximations 87
References 693
Rejecting a design 613
Relating design variables to properties 116
Relating design variables to shape changes 316
Relating finite element analysis model to the design Model 22
Relaxing global tolerance 412
Residual vectors 141
Response calculation 44
Response types 162, 215
Response types FRMASS and COMP 378
Responses
 type-1 39, 44
 type-2 39, 45
 type-3 40, 45
Responses for Topology Optimization 379
Restarting analysis 594
Results
 DSOUG3 523
RMS response 45, 567
RMS response sensitivities 72
S
Searching
 numerical identifying active and violated constraints 75
 Optimum 13
Selecting
 shape change 475
 subcase 456, 483, 511
Sensitivity

analysis 30
coefficients in MSC Nastran 51
computing 51
Sensitivity analysis 2
Setting
 bounds on design variables 498
 display attributes 476
 Isovalue 443
 multiple mass targets 455
 objectives and constraints 509
 Optimization parameters 391, 438, 508
Shape
 basis vector computation 142
 basis vectors 364
 changes over the interior 362
 Optimization of a culvert 343
 sensitivity with rigid elements 59
Shape Optimization 134
Shape Optimization design task 593
Smoothed and remeshed topology 409
Soft convergence decision logic 100
Solution 200 136
Solution for the perturbed displacements 57
Sparse data recovery in sensitivity analysis 72
SPCFORCE 224
Special
 handling for eigenvalue/frequency response 89
 print outputs from creation run 627
 prints for discrete variable Optimization 288
Special prints
 fully stressed design 287
 Topology Optimization 290
Special punch considerations for Topology Optimization 299, 382
Special punch in the case of shape Optimization 299, 369
Special topics 606
Special Topology Optimization settings 244
Square (togex1.dat), example 473
STABDER 233

Static aeroelastic sensitivity analysis 59
Stiffened plate 527
STMOND1 232
STMONP1 232
STOCHAS 206
STRAIN 222
STRESS 224
Structural
 analysis 29
 example 17
 Optimization 22
Subcase
 parameters 510
 spanning responses 139
Submitting a Nastran job with the gmconn keyword 261
Subroutine
 BMTUCD 731
 BMTUGD 725
 BMTUID 719
 BMTUPD 706
 BMXXPD 703
 BRTUCD 730
 BRTUGD 724
 BRTUPD 703
 BRXXPD 703
 BSBRCD 727
 BSBRGB 721
 BSBRID 715
 BSBRPD 700
 BSBRT 712
 BSCON 698
 BSGRQ 711
 BSMSG 733
 MEVBMD 701
 MEVBRD 701
 MSBRCD 728
 MSBRGD 722
 MSBRID 716
 R3SGRT 583

- R3SVALD 251, 584
- Summary
 - Design Optimization 26
 - Superelement Optimization 617
 - Superelements and constraint screening 618
 - Supported analysis disciplines 43
 - Supported superelements 617
 - System cell summary 133
- T
- Table
 - connectivity properties 35
 - design sensitivity response attributes 215
 - element properties 34
 - material properties 35
 - METHOD options 237
 - modified DOPTPRM default for Topology Optimization 244
 - MSCADS options 237
- TACCL 231
- TCHECK and TDMIN 379
- TDISP 231
- Tests for convergence 98
- TFORCE 231
- Theory
 - trust region 611
- Things to consider when designing one-dimensional bending elements 660
- Thinks to know about Optimizers 9
- Three-bar truss 490
- Three-bar truss (tomex1.dat), example 460
- TOMVAR 207
- Topography Optimization 468
- Topology
 - smoothed and remeshed 409
- Topology design
 - glued contact 453
- Topology design variables 115
- Topology Optimization 372
 - glued contact 450
- Topology Optimization input through Patran 427

Topology Optimization variables (TCHECK and TDMIN) 244
Topology Optimization, general 428
Topology Optimization, solution requirements 415
Topometry Optimization 457
Toptomization 478
 example 478, 483
Toptimize for Topometry Optimization 484
TOPVAR 209, 375
 illustration 403
Torsion beam with variations 419
TOTSE 225
Transient dynamic Optimization 574
TRIM 232
Trust region 611
TSPCF 231
TSTRE 231
Turner's problem 513
TVELO 231
Twenty-five bar truss 544
Type-1 properties 36
Type-1 response evaluation 86
Type-1 responses 39, 44
Type-2 properties 38
Type-2 response evaluation 90
Type-2 responses 39, 45
Type-3 properties 38
Type-3 response evaluation 90
Type-3 responses 40, 45
U
 Unformatted design sensitivity 294
Use of design sensitivity and Optimization 3
Use of patran to generate a shape Optimization deck 336
User defined beam libraries 134
Using DRESP1 and DRESP2 entries 123
V
 Vibration of a cantilevered beam (Turner's problem) 513
Viewing final shape 366
Visualization 363
VOLUME 221

Volume response 44
Volume sensitivities 67
W
WEIGHT 220
Weight response 44
Weight sensitivities 67
WMPID Response 44