



GENESIS

DESIGN MANUAL

VERSION 17.0

May 2018

© VANDERPLAATS RESEARCH & DEVELOPMENT, INC.
1767 SOUTH 8TH STREET, SUITE 200
COLORADO SPRINGS, CO 80905
Phone: (719) 473-4611 Fax: (719) 473-4638
<http://www.vrand.com>
email: genesis.support@vrand.com

COPYRIGHT NOTICE

© Copyright, 1991-2018 by Vanderplaats Research & Development, Inc. All Rights Reserved, Worldwide. No part of this manual may be reproduced, transmitted, transcribed, stored in a retrieval system, or translated into any human or computer language, in any form or by any means, electronic, mechanical, magnetic, optical, chemical, manual, or otherwise, without the express written permission of Vanderplaats Research & Development, Inc., 1767 South 8th Street, Suite 200, Colorado Springs, CO 80905.

WARNING

This software and manual are both protected by U.S. copyright law (Title 17 United States Code). Unauthorized reproduction and/or sales may result in imprisonment of up to one year and fines of up to \$10,000 (17 USC 506). Copyright infringers may also be subject to civil liability.

DISCLAIMER

Vanderplaats Research & Development, Inc. makes no representations or warranties with respect to the contents hereof and specifically disclaims any implied warranties of merchantability or fitness for any particular purpose. Further, Vanderplaats Research & Development, Inc. reserves the right to revise this publication and to make changes from time to time in the content hereof without obligation of Vanderplaats Research & Development, Inc. to notify any person or organization of such revision or change.

TRADEMARKS MENTIONED IN THIS MANUAL

GENESIS, Design Studio for Genesis, DOT, BIGDOT and VisualDOC are trademarks of Vanderplaats Research & Development, Inc. NASTRAN is a registered trademark of the National Aeronautics and Space Administration. Other products mentioned in this manual are trademarks of their respective developers or manufacturers.

INTRODUCTION

1

STRUCTURAL OPTIMIZATION
CONCEPTS

2

SHAPE AND SIZING DESIGN
CAPABILITIES

3

SPECIAL SHAPE AND SIZING
DESIGN FEATURES

4

TOPOLOGY OPTIMIZATION

5

EXECUTIVE CONTROL: DESIGN

6

SOLUTION CONTROL: DESIGN

7

SHAPE AND SIZING DESIGN
MODEL DATA

8

TOPOLOGY MODEL DATA

9

DESIGN OUTPUT FILES

REFERENCES

TABLE OF CONTENTS

GENESIS

Design Manual

CHAPTER 1

Introduction

1.1	Structural Optimization Background	3
1.2	General Optimization Features	4
1.3	Review of Analysis Capabilities	6
1.4	Shape ,Sizing, Topography and Topometry Design Capabilities	7
1.5	Topology Design Capabilities	8

CHAPTER 2

Structural Optimization Concepts

2.1	General Optimization	11
2.2	Special Methods for Structural Optimization	15
2.2.1	Constraint Screening	16
2.2.2	Gradient Calculations	17
2.2.3	Approximation Concepts	19
2.2.4	Move Limits During Optimization	21
2.2.5	Convergence to an Optimum	22
2.2.6	Matching Analysis Results	23
2.2.7	Stress Ratio Method	24
2.2.8	The STRDOT Optimizer	26
2.2.9	Discrete Optimization	27
2.2.10	Nonlinear Contact Analysis Optimization	28
2.3	Shape, Sizing Topography and Topometry Design Bulk Data	29
2.4	Topology Design Bulk Data	33

3.1	Basic Sizing Design Capabilities	39
3.1.1	Design Variable Definition (DVAR)	41
3.1.2	Linear Design Variable to Element Property Relationship (DVPROP1)	42
3.1.3	Design Variable Linking (DLINK)	47
3.1.4	Structural Responses (DRESP1)	49
3.1.5	Geometric Responses (DRESPG)	59
3.1.6	Objective Function (DOBJ)	63
3.1.7	Multi-objective Function (DINDEX)	64
3.1.8	Constraints (DCONS and DCONS2)	68
3.1.9	Matching Analysis Results (DMATCH, DMATCH2)	69
3.1.10	Design of Laminated Composites	70
3.2	Basic Shape Design Capabilities	72
3.2.1	Basis Design Approach	73
3.2.2	Anomalies or Unexpected Results in Shape Optimization:	84
3.2.3	Grid Perturbation Approach	86
3.3	Advanced Design Capabilities	102
3.3.1	General Nonlinear Design Variable to Property Relationships (DVPROP2)	103
3.3.2	Element Library Design Variable to Property Relationships (DVPROP3)	105
3.3.3	Design Variable to Composite Property Relationships (DVPROP4)	125
3.3.4	General Nonlinear Response Generation (DRESP2)	128
3.3.5	Constraint Screening (DSCREEN)	132
3.3.6	Move Limits	135
3.3.7	Optimization Process Control (DOPT)	139
3.3.8	Automatic Generation of Element Stress Constraints	143
3.3.9	Automatic Generation of Grid Stress Constraints	147
3.3.10	Automatic Generation of Surface Stress Constraints	149
3.4	Advanced Shape Design Capabilities	150
3.4.1	Coordinate System Definition	151
3.4.2	Move Limits	152
3.4.3	Effect of Nonrectangular Coordinate Systems	156
3.4.4	Automatic Generation of Perturbation Vectors in <i>GENESIS</i>	157

3.4.5	Geometric Perturbation Vectors (DOMAIN and DVGRIDC Data)	160
3.4.6	Geometric Basis Vector (DOMAIN and DVGRIDC Data)	181
3.4.7	Natural Perturbation Vectors (DVSHAPE Data).	182
3.4.8	Natural Basis Vectors (DVSHAPE Data)	186
3.4.9	Grid Basis Vectors (DVBASIS Data)	187
3.5	Reliability Based Optimization - - - - -	191
3.6	Freeform Optimization - - - - -	192
3.6.1	DSHAPE Data Entry	193
3.6.2	Changing Attributes of a Perturbation Vector	194
3.6.3	Freeform with DVGRID data	196
3.6.4	Freeform with DVGRIDC data	198
3.6.5	Defining Fabrication (Symmetry) Constraints for Freeform. 200	
3.6.6	Coarse Freeform	203
3.6.7	Variability Control - Using the GRIDFR Option in Freeform 204	
3.6.8	Freeform Design Variables listed in DSELECT and/or DRESP3 Built- in functions.	205
3.7	Topography Optimization - - - - -	206
3.7.1	Topography Designable Elements	207
3.7.2	Topography Designable Region Selection	208
3.7.3	Topography Basic Shape	209
3.7.4	Defining Fabrication (Symmetry) Constraints for Topography	211
3.7.5	Topography Design Variable.	214
3.7.6	Topography Perturbation Vectors	215
3.7.7	Adding Boundary Grids to the Topography Designable Region	217
3.7.8	Excluding Grids of a Topography Designable Region .	218
3.7.9	Bead Fraction Control - Using the BEADFR Option in Topography	219
3.7.10	Print Equivalent DVAR, DLINK and DVGRID data for Topography Optimization	221
3.8	Topometry Optimization - - - - -	222
3.8.1	How to Set Up Topometry Optimization.	224
3.8.2	Topometry Design Bulk Data	227
3.8.3	Topometry Design Variables.	231
3.8.4	Coarse Topometry Optimization	235

3.8.5	Defining Fabrication (Symmetry) Constraints for Topometry	237
3.8.6	CoarseTopometry with Fabrication Constraints	241
3.8.7	Coarse Topometry with Fabrication Constraints and Selective Design Variables	242
3.8.8	Extended Topometry Regions	243
3.8.9	Extended Topometry Regions with Global Symmetries	245
3.8.10	Extended Topometry with Coase Parameters, Selective Design Variables , Fabrication Constraints and Global Symmetries.....	247
3.8.11	Move Limits in Topometry Optimization	250
3.8.12	Split Responses	251
3.8.13	Using DRESP1 with Topometry Data.....	252
3.8.14	Using DRESP2 with Split DRESP1 and/or Split Design Variables	253
3.8.15	Using User-Supplied Subroutines with DRESP3 with Split DRESP1 and/or Split Design Variables.....	254
3.8.16	Using Built-in Equations in DRESP3 with Split DRESP1 and/or Split Design Variables	255
3.8.17	Topometry Objective Function	256
3.8.18	Topometry Constraints	256
3.8.19	Practical Recommendations.....	256
3.8.20	Topometry Post Processing Result Files	257
3.8.21	Topometry Data Example	258

CHAPTER 4

Special Design Features

4.1	Discrete Design Variables	261
4.1.1	Description of Use	263
4.2	Random Variables for Reliability Analysis	267
4.3	Design Variable Selection	269
4.3.1	Description of Use	270
4.4	Equation Utility	271
4.4.1	Description of Use	272
4.4.2	Use with DVPROP2 data.....	276
4.4.3	Use with DRESP2 and TRESP2 Data.....	278
4.4.4	Error Messages	281
4.5	Adding to the Design Element Library	285
4.5.1	Input Data for User Defined Design Elements (DLIB).	286
4.5.2	User-Supplied Section Property Calculation Interface Function (DLIBPROP)	288

4.5.3	User-Supplied Stress Recovery Calculation Interface Function (DLIB STRESS).....	292
4.6	Using External Analysis Programs -----	296
4.6.1	Capabilities	297
4.6.2	Use of the DRESPU Capability	299
4.6.3	GNUSER: The <i>GENESIS</i> External User Program	300
4.6.4	Use of the DRESP3 / TRESP3 capability	307
4.6.5	DRESP3 Interface Routine for DRESP3 / TRESP3 entries .	310
4.6.6	Example with DRESP3 User Program	311
4.6.7	DRESP3 / TRESP3 Built-in Functions	314
4.7	Shifted Responses -----	316
4.7.1	Overview.....	318
4.7.2	Data Entry.....	319
4.7.3	Loading Frequency Dependent Constraint Bounds....	320
4.8	Restart Capability -----	321
4.8.1	Restart from any Previous Design Cycle.....	322
4.9	Mode Tracking -----	323
4.10	Matching Measured Data -----	325
4.11	Matching Eigenvector Components -----	327
4.12	Relative Constraint Bounds -----	328
4.13	Allowable Probability of Failure on Constraint Entries	329
4.14	Mesh Smoothing -----	330
4.15	Stress Ratio -----	331
4.16	Customization Through Scripts -----	333
4.16.1	Lua scripting engine.....	333

CHAPTER 5

Topology Optimization

5.1	Introduction -----	349
5.2	Topology Design Bulk Data -----	350
5.2.1	Topology Designable Elements.....	351
5.2.2	Topologically Designable Region Selection	352
5.2.3	Relationships Between Design Variables and Material Properties	354
5.2.4	Move Limits in Topology Optimization	357
5.2.5	Fabrication Constraints	358

5.2.6	Extended Topology Regions	359
5.2.7	Cloned Topology Regions	360
5.2.8	Selecting Responses.	365
5.2.9	Topology Design Objective Function	372
5.2.10	Topology Constraints.	377
5.2.11	Defining the Symmetry Planes for Fabrication Constraints	378
5.2.12	Defining the Fabrication Constraints.	380
5.2.13	Combining Fabrication Constraints.	393
5.2.14	Topology Extra Variables	396
5.2.15	General Nonlinear Response Generation (TRESP2)	397
5.2.16	Topology Variable Selection	400
5.2.17	Progressive Rule	402
5.2.18	Hybrid Based Topology Method	405
5.2.19	Fully Polarized Results	406
5.2.20	Use of Optimization Parameters	407
5.3	Topology Post Processing Result Files - - - - -	409
5.4	Using the AUTORIB capability with Topology Optimization 410	
5.5	Practical Recommendations - - - - -	412
5.5.1	Do Not Mix Load Combinations with Topology Optimization	413
5.6	Example - - - - -	414
5.7	Mixing Topology with Shape, Sizing, Topometry, Topography and/or Freeform Optimization - - - - -	416
5.7.1	Mixing Topology and Sizing Optimization	417
5.7.2	Mixing Topology and Topometry Optimization	418
5.7.3	Mixing Topology and Shape, Topography and/or Freeform Optimization	419
5.7.4	Mixing Topology and Parametric Optimization: Mass versus Mass Fraction.	420

CHAPTER 6

Executive Control

6.1	ANALYSIS - - - - -	427
6.2	DLIB - - - - -	428
6.3	DRESP3 - - - - -	430
6.4	GNUSER - - - - -	431
6.5	HIS - - - - -	432

6.6	RESTART	433
6.7	RST	434
6.8	SCRIPT	435
6.9	SENSITIVITY	436
6.10	SSOLID	437
6.11	TOPOLOGY	438
6.12	TSURFACE	439

CHAPTER 7

Solution Control

7.1	Natural (Displacement) Basis (Perturbation) Vectors	443
7.2	Grid Basis Vectors	444
7.3	General Design Output Control Commands	445
7.4	Shape, Sizing, Topography, Topometry and Freeform Output Control Commands	446
7.5	Topology Output Control Commands	448
7.6	Output Selection	449
7.7	Solution Control Data	451
7.7.1	APRINT	452
7.7.2	AUTORIB	453
7.7.3	BASIS	455
7.7.4	DENSITY	456
7.7.5	DESIGN	457
7.7.6	DPRINT	458
7.7.7	DRESP2	459
7.7.8	DVBASIS	460
7.7.9	DVGRID	461
7.7.10	DVSHAPE	462
7.7.11	GRAPH	464
7.7.12	KAASENS	465
7.7.13	MAASENS	466
7.7.14	MODTRK	467
7.7.15	MPRINT	468
7.7.16	OPRINT	469
7.7.17	PERTURBATION	470
7.7.18	RELIABILITY	471
7.7.19	SENSITIVITY	472

7.7.20	SHAPE	473
7.7.21	SIZING	474
7.7.22	SSOL	475
7.7.23	THICKNESS	476
7.7.24	TSURF.....	477
7.7.25	TVAR	478
7.7.26	UPRINT	479

CHAPTER 8

Shape, Sizing, Topography, Topometry and Freeform Design Model Data

8.1	Shape and Sizing Design Data Relationships -----	483
8.2	Shape, Sizing, Topography, Topometry and Freeform Design Bulk Data -----	484
8.2.1	DCONS	485
8.2.2	DCONS2	487
8.2.3	DEQATN.....	489
8.2.4	DINDEX	492
8.2.5	DLIB	494
8.2.6	DLINK.....	496
8.2.7	DMATCH	497
8.2.8	DMATCH2	499
8.2.9	DOBJ	501
8.2.10	DOMAIN.....	502
8.2.11	DOPT	507
8.2.12	DRESP1.....	522
8.2.13	DRESP2.....	554
8.2.14	DRESP3.....	558
8.2.15	DRESPG	564
8.2.16	DRESPU	569
8.2.17	DSCREEN.....	570
8.2.18	DSELECT	572
8.2.19	DSHAPE	574
8.2.20	DSHIFT	579
8.2.21	DSPLIT	582
8.2.22	DTABLE	588
8.2.23	DTGRID	589
8.2.24	DVAR	596
8.2.25	DVGRID	598
8.2.26	DVGRIDC.....	600
8.2.27	DVPROP1	603

8.2.28	DVPROP2	612
8.2.29	DVPROP3	621
8.2.30	DVPROP4	663
8.2.31	DVSET	665
8.2.32	DVSET1	666

CHAPTER 9

Topology Design Model Data

9.1	Topology Data Relationships	671
9.2	Topology Bulk Data	673
9.2.1	DEQATN	674
9.2.2	DOPT	677
9.2.3	DSCREEN	687
9.2.4	DSHIFT	689
9.2.5	DTABLE	692
9.2.6	TCONS	693
9.2.7	TCONS2	695
9.2.8	TCYCLE	697
9.2.9	TINDEX	699
9.2.10	TOBJ	702
9.2.11	TPROP	703
9.2.12	TPROPC	706
9.2.13	TRESP1	708
9.2.14	TRESP2	720
9.2.15	TRESP3	724
9.2.16	TSELECT	729
9.2.17	TSYM1	732
9.2.18	TSYM2	741
9.2.19	TSYM3	750
9.2.20	TVAR	759

CHAPTER 10

Multi-Model Master Data

10.1	Multi-Model Optimization	763
10.2	Master Data Executive Control	764
10.2.1	\$	765
10.2.2	ANALYSIS	766
10.2.3	CEND	767
10.2.4	CHECK	768
10.2.5	DIAG	769
10.2.6	DRESP3	770

10.2.7	INCLUDE	771
10.2.8	IOBUFF	772
10.2.9	LENVEC	773
10.2.10	MODEL	774
10.2.11	RESTART	775
10.2.12	SCRIPT	776
10.2.13	SENSITIVITY	777
10.3	Master Data Solution Control -----	778
10.3.1	APRINT	779
10.3.2	BEGIN BULK	780
10.3.3	DPRINT	781
10.3.4	DRESP2	782
10.3.5	ECHO	783
10.3.6	ECHOON	785
10.3.7	ECHOOFF	786
10.3.8	INCLUDE	787
10.3.9	LINE	788
10.3.10	OPRINT	789
10.3.11	SUBTITLE	790
10.3.12	TIMES	791
10.3.13	TITLE	792
10.3.14	UPRINT	793
10.4	Master Data Bulk Data -----	794
10.4.1	DCONS	795
10.4.2	DCONS2	796
10.4.3	DEQATN	797
10.4.4	DINDEX	800
10.4.5	DOBJ	802
10.4.6	DOPT	803
10.4.7	DRESP2	804
10.4.8	DRESP3	806
10.4.9	DSCREEN	810
10.4.10	DTABLE	811
10.4.11	ENDDATA	812
10.4.12	INCLUDE	813
10.4.13	TCONS	814
10.4.14	TCONS2	815
10.4.15	TINDEX	816
10.4.16	TOBJ	818
10.4.17	TRESP2	819
10.4.18	TRESP3	821

CHAPTER 11**Output Files**

11.1 Summary of <i>GENESIS</i> Design Files	827
11.2 Program Design Output	828
11.3 Design Cycle History File (pname.HIS)	829
11.4 Design Information File (pname.OPT)	830
11.5 Updated Input File (pnameUPDATExx.dat)	831
11.6 Graph File (pname.html or pname.ps)	832
11.7 Sensitivity Post-Processing Data (pnamexx.SEN)	833
11.8 Perturbation Post-Processing Data (pname.DVG)	837
11.9 Shape Post-Processing Data (pname.SHP)	838
11.10 Natural Basis (Perturbation) Vector (pname.DVS)	839
11.11 Grid Basis Vector (pname.DVB)	840
11.12 AUTORIB Output File (pname.RIB)	841
11.13 Topology Density File (pname.DNS)	842
11.14 Topology Density File (pnameDENSxx.ext)	843
11.15 Topology Isodensity File (pnameTSURFxx.dat)	844
11.16 Sizing Post-Processing Data (pnameOPOSTxx.ext)	845
11.17 Shell to Solid Results File (pnameSSOLxx.dat)	847
11.18 Sensitivities of Guyan Reduced Stiffness Matrix (pnameexxyy.SKA or pnameexxyyyyyyyyyy.SKA)	848
11.19 Sensitivities of Guyan Reduced Mass Matrix (pnameexxyy.SMA or pnameexxyyyyyyyyyy.SMA)	849
11.20 Element Reliability Results File (pnameRELxx.pch)	850

CHAPTER 12**References**

12.1 References	853
-----------------	-----

CHAPTER A**Diagnostic Information**

1.1 Diagnostic Information: Design	857
1.2 The DIAG Command	858

CHAPTER 1

Introduction

- Structural Optimization Background
- General Optimization Features
- Shape ,Sizing, Topography and Topometry Design Capabilities
- Topology Design Capabilities

1.1 Structural Optimization Background

Structural optimization methods, where a general finite element analysis capability is combined with formal nonlinear numerical optimization, dates from 1960 when Professor L. A. Schmit [1] developed the basic concepts of structural synthesis. The original methods were quite general, but required that a large number of designs be evaluated using the finite element analysis. A major breakthrough was made in 1974-1976 when Schmit and Farshi [2] and Schmit and Miura [3] developed the general framework for structural optimization using what has come to be called “approximation concepts.” Using this approach, a high quality approximation to the original problem is created which can be used in optimization. By repetitively applying these approximations, an optimum design could be achieved at much lower cost.

Beginning in the mid-1980's, most major finite element analysis program vendors have incorporated optimization into their codes. Some used very simple approaches, based on 1960 and early 1970 technology, while others used the approximation techniques of the mid-1970's. None use current technology.

However, in the 1980's, and still today, the technology of structural optimization has continued to progress with the development of what is called “second generation approximation techniques.” Examples include the force approximation method for stress constraints, developed by Vanderplaats and Salajegheh [4] and the Rayleigh quotient approximation, developed by R. Canfield [5]. These methods require a close coupling between the finite element analysis and the optimization capabilities. Consequently, it is difficult and costly to adapt existing finite element analysis programs to the latest technology. Therefore, it has become necessary to consider the development of a structural optimization code which is created, from the start, to be a design program. Such software must be a design program with the necessary finite element analysis available, in contrast to a finite element analysis code which has optimization added later.

GENESIS is just such a program. While this program contains an excellent finite element analysis capability, the principal motive is to provide a design code. Thus, the analysis features have been created to support the design process, rather than having design added later. This has allowed VR&D to create a premier design optimization capability, while still providing a high quality analysis option as the basis for the design decisions.

1.2 General Optimization Features

The *GENESIS* program will solve both member dimension (sizing, topometry) and coordinate location (shape, topography) optimization problems. A wide range of options are available to define the design problem in an efficient manner. The user interface is very similar to the NASTRAN [6] input data, so users who are familiar with that program will find it easy to use *GENESIS* or to modify their NASTRAN data for use by here.

GENESIS can also solve the material allocation (topology) optimization problem. Topology optimization is used by engineers to get a preliminary or conceptual design for further refinement using sizing and shape optimization.

GENESIS is written to operate on everything from personal computers to supercomputers. This allows the user to solve a wide variety of everyday design tasks at his desk, while sending only the largest problems to a mainframe or supercomputer. To make the best use of computer resources, design problems of significant size may be solved using workstations at night, when they normally stand idle.

The basic building blocks are a main control program and modules for analysis and gradient computations, creation of high quality approximations for use in optimization, and the optimizer itself. A very general concept of the overall program structure is shown in **Figure 1-1**.

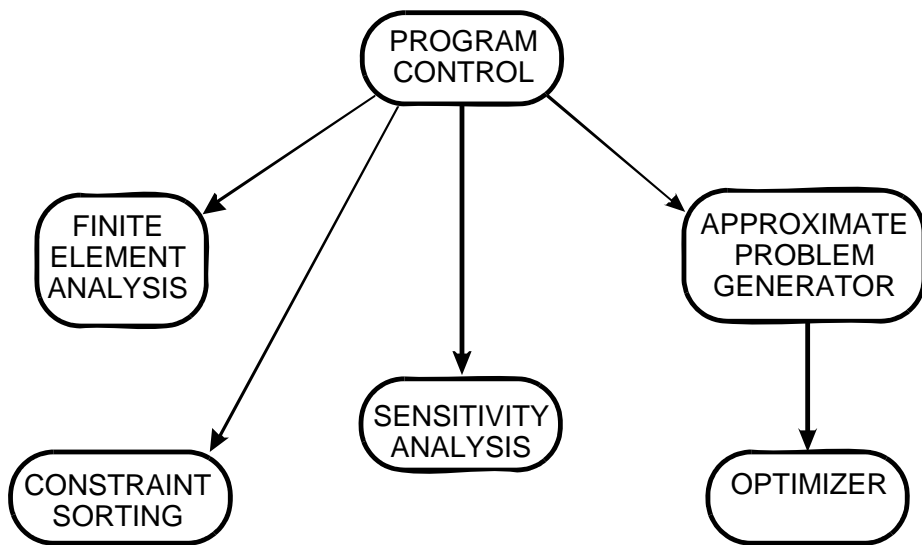


Figure 1-1 Basic Building Blocks

The Control module acts as the program director. The control module calls an input module which reads and organizes all data. Next, the FEM analysis module is called to perform a complete analysis and evaluate the responses of interest. Then a module is called which evaluates all design constraints and sorts them according to degree of criticality. Only those constraints that are critical or potentially critical are retained for consideration during the current design cycle. The FEM Sensitivity module is then called to evaluate the gradients of retained responses with respect to design variables. The formal optimization problem is then created and solved by calling the Approximate Problem Generator. This module, in turn, calls the DOT or BIGDOT optimizer to solve the approximate problem. During this process, the design objective and constraint functions are evaluated as functions of the design variables. As can be seen, the overall optimization process is far more complex than simply coupling the FEM analysis with an optimizer. However, the result of this complexity (which the user does not see) is a very efficient and robust design program. Thus, using *GENESIS*, an optimum design can be found as economically as an only “acceptable” design is found using traditional methods.

1.3 Review of Analysis Capabilities

GENESIS will solve analysis problems in static, vibration and dynamic linear elasticity where the structure is modeled as an assemblage of rod, beam, bending/membrane, shear, composite, scalar and solid elements. Multiple loading and multiple boundary conditions are considered. Responses that are calculated include internal forces, stresses, strains, joint displacements, velocities, accelerations, grid point stresses, reaction forces, system strain energies, mass, volume, system moments of inertia and vibration frequencies. Single and multipoint constraints are allowed.

Inertia relief is available for static analysis, and it can be used simultaneously with different support conditions.

Guyan reduction is available for vibration analysis and it can be used simultaneously with different boundary (ASET) conditions.

Buckling analysis is also available for checking the stability of the structure subject to statics loads.

GENESIS will also solve linear static heat transfer problems with heat flux, conduction, convection and adiabatic boundary conditions. Volumetric heat generation loads are available. Grid point temperatures and reaction fluxes are calculated. The resulting thermal loads can be automatically applied to a linear static structural analysis.

Two linear equation solvers are available; sparse matrix and skyline. The sparse matrix solver is normally used as the default. It is not available at all installations. In this case, the skyline solver is used. The user can choose the solver to be used by specifying the analysis parameter "SOLVER".

Modules are also included to calculate gradients (sensitivities) of responses with respect to the design variables or to intermediate variables such as section properties. This feature is used by the design module as part of the optimization process. The gradients of the retained responses with respect to the independent design variables are available to the user.

1.4 Shape ,Sizing, Topography and Topometry Design Capabilities

Extensive design capabilities are available. Whether you wish to design a simple truss, a complicated three dimensional solid structure or a structure made up of a variety of element types, *GENESIS* will modify both member sizes and geometrical shape in search of an optimum design.

Design variables may be individual member dimensions and/or grid locations, or may be linear or nonlinear combinations of these. Dimension or shape variables may be linked to maintain symmetry and manufacturability. Basis vectors are used to create smooth variations in structural dimensions and shapes. For designing frame or plate structures, a library of common elements is available to simplify the task. Also, the user may provide his own subroutine to calculate frame section properties as functions of the design variables or he/she may use “synthetic” functions to define the section properties at run time.

Any response that is calculated, such as mass, a stress or displacement, temperature or vibration frequency, may be chosen as the design objective to be minimized or maximized. At the same time, you may impose limits (constraints) on any responses you choose. Additionally, you may create “synthetic” functions of the design variables and other responses and these functions may be the objective or may be constrained. This provides a versatile means of incorporating proprietary design requirements into the optimization process or for imposing requirements that are not already included in the program, but which are functions of the variables and responses that are available in the program. Finally, you may adjust the analysis model to match experimental results. For example, you may change member dimensions or geometry so the analysis will produce a displacement or frequency which you have measured in a structural test.

1.5 Topology Design Capabilities

With topology optimization, regions of the structure that have the least contribution to the overall stiffness or natural frequency are identified. This tells the user which regions should be removed from the structure to minimize the mass while having the least impact on the performance of the structure. Properties are selected to be designed, and design variables are automatically created to control the stiffness and density of every element in that property. Any property that references MAT1, MAT4, MAT5, MAT9 or MAT11 material data may be designed. In addition, any PCOMP or PCOMPG property can be designed. Available responses include static displacements, strain energies, natural frequencies, buckling load factors, direct and modal displacements, velocities and accelerations; random root mean square displacement, velocities and accelerations, inertia responses and mass fraction. In addition, heat transfer compliance (HTC) and temperature responses can be used in topology. Special features for topology optimization include data to enforce symmetries and/or fabrication constraints in the structure and a capability to reduce “checkerboard” like results. Mode tracking is available for both frequencies and buckling load factors.

Topology results, for post processing, are smoothed and surfaces representing density levels are output to aid the user to interpret topology optimization results.

CHAPTER 2

Structural Optimization Concepts

- General Optimization
- Special Methods for Structural Optimization
- Shape, Sizing Topography and Topometry Design Bulk Data
- Topology Design Bulk Data

2.1 General Optimization

Numerical optimization methods provide a uniquely general and versatile tool for design automation. Research and applications to structural design has been extensive and today these methods are finding their way into engineering offices. The methods that form the basis of most modern optimization were developed roughly 30 years ago, and the first application to nonlinear structural design was presented by Schmit in 1960 [1]. Much of the research in structural design since 1975 has been devoted to creating methods that are efficient for structural design problems where the analysis is expensive. This has resulted in various approximation methods that allow a high degree of efficiency while maintaining the essential features of the original problem.

Here, we will first define the general design task in terms of optimization. For structural optimization, we create an approximation to the original problem. This approximate problem is solved by the optimizer. The advantage is that it is not necessary to repeatedly call the finite element analysis during the actual optimization process. This greatly reduces the overall cost of structural design. In *GENESIS*, the DOT [7] or BIGDOT optimization programs, developed by VR&D, are used to solve this optimization sub-problem.

In Section 2.2, some of the special techniques that have been devised to make structural optimization efficient will be discussed. It is these special techniques, that are contained in *GENESIS*, which make modern structural optimization efficient, relative to other applications.

Mathematical programming (the formal name for numerical optimization) provides a very general framework for scarce resource allocation, and the basic algorithms originate in the operations research community. Engineering applications include structural design, chemical process design, aerodynamic optimization, nonlinear control system design, mechanical component design, multi-discipline system design, and a variety of others. Because the statement of the numerical optimization problem is so close to the traditional statement of engineering design problems, the design tasks to which it can be applied are inexhaustible.

In the most general sense, numerical optimization solves the nonlinear, constrained problem: Find the set of design variables, X_i , $i = 1, N$ contained in vector X , that will

$$\text{Minimize } F(\mathbf{X}) \quad (\text{Eq. 2-1})$$

Subject to;

$$g_j(\mathbf{X}) \leq 0 \quad j = 1, M \quad (\text{Eq. 2-2})$$

$$X_i^L \leq X_i \leq X_i^U \quad i = 1, N \quad (\text{Eq. 2-3})$$

Equation **(Eq. 2-1)** defines the objective function, which depends on the values of the design variables, \mathbf{X} . **(Eq. 2-2)** is inequality constraints. Equality constraints of the form $h_k(\mathbf{X}) = 0 \quad k = 1, L$ could also be included. This is achieved by using two equal, but opposite in sign, inequality constraints. **(Eq. 2-3)** defines the region of search for the minimum. This provides limits on the individual design variables. The bounds defined by **(Eq. 2-3)** are referred to as side constraints. A clear understanding of the generality of this formulation makes the breadth of problems that can be addressed apparent. However, there are some important limitations to the present technology. First, it is assumed that the objective and constraint functions be continuous and smooth (continuously differentiable). Experience has shown this to be a more theoretical than practical requirement and this restriction is routinely violated in engineering design. A second requirement is that the design variables contained in \mathbf{X} be continuous. In other words, we are not free to choose structural sections from a table. Also, we cannot treat the number of plies in a composite panel as a design variable, instead treating this as a continuous variable and rounding the result to an integer value. It is not that methods do not exist for dealing with discrete values of the variables. It is just that available methods lack the needed efficiency for widespread application to real engineering design. VR&D has developed software to deal with this case, and this is expected to be added in the future. Finally, even though there is no theoretical limit to the number of design variables contained in \mathbf{X} , if we use optimization as a “black box” where we simply couple an analysis program to an optimization program, the number of design variables that can be considered is limited to the order of fifty. Again, there are many exceptions to this, but it is a conservative general rule. Also, this is not too great a restriction when we recognize that using graphical methods would limit us to only a few design variables. Furthermore, using the special techniques contained in *GENESIS*, the number of design variables may be in the hundreds.

The general problem description given above is remarkably close to what we are accustomed to in engineering design. For example, assume we wish to determine the dimensions of a structural member that must satisfy a variety of design conditions. We normally wish to minimize mass, so the objective function, $F(\mathbf{X})$, is the mass of the structure, which is a function of the sizing variables. However, we also must consider constraints on stresses, deflections, buckling and perhaps dynamic response limits. Assuming we model the structure as an assemblage of finite elements, we can calculate the stresses in the elements under each of the prescribed loading conditions. Then a typical stress limit may be stated as

$$\sigma^L \leq \sigma_{ijk} \leq \sigma^U \quad (\text{Eq. 2-4})$$

where i = element number, j = stress component and k = load condition. The compression and tensile stress limits are σ^L and σ^U , respectively (if we use a von Mises stress criterion, only σ^U would be used). While **(Eq. 2-4)** may initially appear to be different from the general optimization statement, it is easily converted to two equations of the form of **(Eq. 2-2)** as

$$g_1(\mathbf{X}) = \frac{\sigma^L - \sigma_{ijk}}{|\sigma^L|} \leq 0 \quad (\text{Eq. 2-5})$$

$$g_2(\mathbf{X}) = \frac{\sigma_{ijk} - \sigma^U}{|\sigma^U|} \leq 0 \quad (\text{Eq. 2-6})$$

Thus, the formal statement of the optimization task is essentially identical to the usual statement of the structural design task.

The denominator of **(Eq. 2-5)** and **(Eq. 2-6)** represent normalization factors. This is important since it places each constraint in an equal basis. For example, if the value of a stress constraint is -0.1 and the value of a displacement constraint is -0.1, this indicates that each constraint is within 10% of it's allowable value. Without normalization, if a stress limit is 20,000, it would only be active (within a value of 0.1) if it's value was 19,999.9. This is not meaningful since loads, material properties, and other physical parameters are not known to this accuracy.

It is often assumed that for optimization to be used, the functional relationships must be explicit. However, this is categorically untrue. It is only necessary to be able to evaluate the objective and constraint functions for proposed values of the design variables, \mathbf{X} .

Using optimization as a design tool has several advantages. We can consider large numbers of variables relative to traditional methods. Also in a new design environment, we may not have a great deal of experience to guide us and so optimization often gives unexpected results which can greatly enhance the final product. Finally, one of the most powerful uses of optimization is to make early design trade-offs using simplified models. Here we can compare optimum designs instead of just comparing point designs.

On the other hand, optimization has some disadvantages to be aware of. The quality of the result is only as good as the underlying analysis. In the case of finite element analysis, we must remember that this is just a model of the real structure. Thus, if we ignore or forget an important constraint, optimization will take advantage of it, leading to a meaningless if not dangerous design. Also, there is a danger that by optimizing we will reduce the hidden factors of safety that now exist. In this context, we should re-think our use of optimization, using it as a design tool and not as a sole means to an end product.

However, assuming we agree that optimization is useful, it is also important to know how these algorithms solve our design problems. Next, we briefly outline the basic optimization process, contained in the DOT program, to provide some insight into the numerical techniques used.

Most optimization algorithms do just what good designers do. They seek to find a perturbation to an existing design which will lead to an improvement. Thus, we seek a new design which is the old design plus a change, so

$$\mathbf{X}^{\text{new}} = \mathbf{X}^{\text{old}} + \delta \mathbf{X} \quad (\text{Eq. 2-7})$$

Optimization algorithms use much the same formula, except it becomes a two step process. Here we update the design by the relationship

$$\mathbf{X}^{q+1} = \mathbf{X}^q + \alpha \mathbf{S}^q \quad (\text{Eq. 2-8})$$

where $\alpha \mathbf{S}^q$ in **(Eq. 2-8)** is equivalent to $\delta \mathbf{X}$ in **(Eq. 2-7)**. Here q is the iteration number, and numerous search directions (iterations) will be needed to reach the optimum. The engineer must provide an initial design, \mathbf{X}^0 , but it need not be feasible; it may not satisfy the inequalities of **(Eq. 2-2)**. Optimization will then determine a “Search Direction,” \mathbf{S}^q that will improve the design. If the design is initially feasible, the search direction will reduce the objective function without violating the constraints. If the initial design is infeasible, the search direction will point toward the feasible region, even at the expense of increasing the objective function.

The next question is how far can we move in direction \mathbf{S}^q before we must find a new search direction. This is called the “One-Dimensional Search” since we are just seeking the value of the scalar parameter, α , to improve the design as much as possible. If the design is feasible and we are reducing the objective function, we seek the value of α that will reduce $F(\mathbf{X})$ as much as possible without driving any $g_j(\mathbf{X})$ positive or violating any bound (side constraint) on the components of \mathbf{X} . If the design is initially infeasible, we seek the value of α that will overcome the constraint violations if possible, or will otherwise drive the design as near to the feasible region as possible. Note that this is precisely what a design engineer does under the same conditions. The difference is that optimization does it without the need to study many pages (or screens full) of computer output.

There are a wide variety of algorithms for determining the search direction, \mathbf{S} , as well as for finding the value of α [8]. Determining α is conceptually a simple task. For example, we may pick several values of α and calculate the objective and constraint functions. Then we fit a polynomial curve to each function and determine the value that will minimize $F(\mathbf{X})$ or drive $g_j(\mathbf{X})$ to zero. Since we picked a search direction that will improve the design, we need only consider positive values of α . The minimum positive value of α from among all of these curve fits is the one we want. For further details of how the optimization problem is solved, the DOT user’s manual [7] and reference [8] may be useful.

2.2 Special Methods for Structural Optimization

The state of the art in structural optimization is now reasonably well developed, relative to other applications. To provide an overview of the methods used in *GENESIS*, we will briefly outline several key ingredients to the structural optimization process. The key concept is that we solve the structural design problem without the large number of full finite element analyses that would be required if we simply coupled the FEM and Optimizers. It is important to understand, however, that even though we use approximations to achieve this, we retain the key features of the detailed analysis model. Thus, when we are finished, we will have the same design we would find if we were able to use the FEM analysis directly during optimization.

The basic optimization process contained in *GENESIS* is summarized in the following 10 steps.

1. Preprocess all input data and perform all non-repetitive operations (e.g., check data for correctness, create internal tables, set up the overall program flow).
2. Perform a detailed finite element analysis for the initial proposed design. Evaluate the design objective and all constraints.
3. Screen all constraints and retain those that are critical or near critical for further consideration. Typically only $2n$ to $3n$ constraints are retained, where n is the number of independent design variables.
4. Perform the sensitivity analysis (gradient computations) for the responses included in the objective function and the retained constraints.
5. Set up a high quality approximation to the original problem and solve it using the DOT or BIGDOT optimizer.
6. If no design improvement is possible, exit. This is called “soft convergence.”
7. Assuming the design variables have changed, update the analysis data and perform a detailed finite element analysis for this proposed design.
8. Evaluate the precise objective function and all constraints.
9. If the design is not improving and if all constraints are satisfied within a specified tolerance, exit. This is called “hard convergence.”
10. If progress is still being made toward the optimum, we say that one “design cycle” has been completed. We then repeat the process from step 3.

From this brief outline, it is seen that key components of the structural optimization process include the finite element analysis, constraint screening, sensitivity analysis, creation and solution of the approximate optimization problem, and judging when convergence has been achieved. It is assumed that the reader is familiar with the analysis process. The key parts of the design optimization process will be briefly discussed here to give a general understanding of the *GENESIS* design capabilities.

2.2.1 Constraint Screening

The first thing to note is that the design of real structures requires that a very large number of constraints must be satisfied. For example, assume we model a large structure with hundreds or even thousands of finite elements. We may recover the stress at several locations in each element under many different loading conditions. Assume we must limit the von Mises stress to be less than or equal a specified value, σ_a . Then, $\sigma_{ijk} \leq \sigma_a$ where i = the element number, j = the stress recovery location and k = the load condition. Clearly, the combination ijk can become very large, often over one million.

Because the optimization process requires gradients of the constraints, this could lead to a very costly design sensitivity process, far exceeding the cost of a single analysis. Therefore, we first “screen” the constraints and retain only those that are critical or potentially critical for the current design cycle. This is a two step process. First, we delete all constraints that are more negative than, say -0.3 (not within 30% of being critical). Next, we search the set of retained constraints and further delete all but a specified subset in a given region of the structure. The reason for this is that many nearby points in the structure may have approximately the same stress. However, only a few of these stress responses need to be retained to direct the design process.

2.2.2 Gradient Calculations

Having identified the responses that will be retained during the current design cycle, the next step is to evaluate their gradients (sensitivities). The sensitivity of a static response, R , (e.g., stress, displacement, strain energy) with respect to a design variable, X , is determined by the chain rule of differentiation as follows:

$$\frac{dR}{dX} = \frac{\partial R}{\partial X} + \frac{\partial R}{\partial U} \frac{\partial U}{\partial X} \quad (\text{Eq. 2-9})$$

Using the governing global equilibrium equations ($[K]U = F$) the displacement sensitivities are determined as:

$$\frac{\partial U}{\partial X} = [K]^{-1} \left\{ \frac{\partial F}{\partial X} - \left[\frac{\partial K}{\partial X} \right] U \right\} \quad (\text{Eq. 2-10})$$

where $\left\{ \frac{\partial F}{\partial X} - \left[\frac{\partial K}{\partial X} \right] U \right\}$ are referred to as pseudo-loads.

Therefore, the response sensitivity becomes:

$$\frac{dR}{dX} = \frac{\partial R}{\partial X} + \frac{\partial R}{\partial U} [K]^{-1} \left\{ \frac{\partial F}{\partial X} - \left[\frac{\partial K}{\partial X} \right] U \right\} \quad (\text{Eq. 2-11})$$

The direct method first calculates the displacement sensitivity $\frac{\partial U}{\partial X}$ and uses that to

calculate $\frac{\partial R}{\partial U} \frac{\partial U}{\partial X}$ to form the response sensitivity. This method requires performing a forward/back substitution (i.e., equivalent to solving a static loadcase) for each design variable.

The adjoint method first calculates $[K^{-1}]^T \left\{ \frac{\partial R}{\partial U} \right\}^T$ and then dots that with the pseudo-

load to form the second part of the response derivative. Note that because K is symmetric, $[K^{-1}]^T = [K]^{-1}$. Therefore, this method require one forward/back substitution for each response. If the number of retained responses is smaller than the number of design variables, then the adjoint method should provide better performance.

The sensitivity method is selected in *GENESIS* by the optimization parameter, **ADJOIN**. A value of 0 selects the direct method, while a value of 1 selects the adjoint method. By default, *GENESIS* will select the most efficient method automatically.

Structural Optimization Concepts

The pseudo-loads are formed on an element-by-element basis and assembled into a global vector. Where possible and efficient, an exact analytical process is used throughout the sensitivity calculations. In other cases, a semi-analytic technique is used, whereby the pseudo-load is calculated by finite difference, but the remainder of the sensitivity calculations are fully analytic.

2.2.3 Approximation Concepts

The key to efficiency of modern structural optimization lies in what is called “approximation concepts”. The simplest approximation method would be to create a linear approximation to the objective and constraint functions as;

$$\tilde{F}(\mathbf{X}) = F(\mathbf{X}^0) + \nabla F(\mathbf{X}^0) \cdot \{\mathbf{X} - \mathbf{X}^0\} \quad (\text{Eq. 2-12})$$

$$\tilde{g}_j(\mathbf{X}) = g_j(\mathbf{X}^0) + \nabla g_j(\mathbf{X}^0) \cdot (\mathbf{X} - \mathbf{X}^0) \quad j = 1, M \quad (\text{Eq. 2-13})$$

These approximations are then sent to the optimizer to modify the design. In practice, move limits are imposed on the design variables so that they are not changed beyond the region of applicability of the approximation.

The repeated application of simple linearizations such as this is called “Sequential Linear Programming” and this has been used for nearly 30 years as a valid optimization strategy. However, in the special case of structural design, we are able to create approximations that are valid over a much wider range of the design variables.

Consider calculating the stress in a simple rod element, as $\sigma = \frac{F(A)}{A}$. Now, if we linearize the stress in terms of the design variable, A, we get

$$\tilde{\sigma} = \sigma^0 + \frac{\partial \sigma}{\partial A} \delta A = \sigma^0 + \frac{1}{A^0} \left\{ \frac{\partial F^0}{\partial A} - \sigma^0 \right\} \delta A \quad (\text{Eq. 2-14})$$

The equation $\sigma = \frac{F}{A}$ is quite nonlinear in A, and so the approximation is valid only for small changes in A. Now consider using an “intermediate” variable, $X=1/A$, so $(\sigma = FX)$. Linearizing with respect to X gives

$$\tilde{\sigma} = \sigma^0 + \frac{\partial \sigma}{\partial X} \delta X = \sigma^0 + \left\{ F^0 + \frac{\partial F^0}{\partial X} X^0 \right\} \delta X \quad (\text{Eq. 2-15})$$

The equation $\sigma = F(X) \cdot X$ is more linear in X and, in the special case of a statically determinate structure F is independent of X, so the approximation is precisely linear in X. The optimizer will now treat X as the design variable. When the approximate optimization is complete, we recover the cross-sectional areas as $A=1/X$.

As another example, consider a rectangular beam element, with the width, B, and height, H, as design variables. We can treat the section properties (e.g., A, I, J) as intermediate variables and calculate the approximate stresses and displacements in terms of these. Then, when the optimizer requires the stress or displacement values, we

first calculate the section properties as explicit functions of the design variables, B and H, and recover the responses from the linearized quantities. In this fashion, we retain considerable nonlinearity contained in the design variable to section property relationships.

Now take this process one step further by considering intermediate responses. Here, for stress constraints in a rod, we approximate the force, F, instead of stress in the rod. When stress is needed, we first calculate the approximate force in the element as

$$\tilde{F} = F^0 + \frac{\partial F}{\partial A} \delta A \quad (\text{Eq. 2-16})$$

Then we recover the stress as $\sigma \approx \frac{\tilde{F}}{A}$. This can be shown to be a higher quality approximation than found by using the reciprocal variable, X, and a much higher quality approximation than found by direct linearization.

GENESIS uses these approximations, as well as a variety of others, to improve the overall efficiency and reliability of the structural design process. The key concept is that we are free to restate the optimization problem in whatever form is best, as long as we retain the important mathematical features of the original problem. By using high quality approximations, we use only the approximated functions during optimization. In this way, we avoid the large number of finite element analyses normally needed for optimization using numerical search methods.

2.2.4 Move Limits During Optimization

Even though *GENESIS* uses very high quality approximations to drive the design process, they are still not a precise representation of the model analyzed by the FEM model. Therefore, it is important to limit the design changes during any single design cycle. To do this, “move limits” are used, being the amount by which the design variables can change before it is considered necessary to perform a detailed analysis of the new proposed design. Additionally, because *GENESIS* uses intermediate variables, move limits are imposed on the element section properties, since these are used in the Taylor series expansions. Typically, the design variables and section properties are allowed to change by 50% during a design cycle. This is roughly 4-5 times the design changes that could be allowed if simple linearization methods were used. In practice, the move limits are not active as the optimization process converges, but they are important in the early design cycles to properly direct the design process.

2.2.5 Convergence to an Optimum

Because the optimization process is iterative, it is necessary to judge when the process is complete and should be stopped. *GENESIS* uses a variety of mechanisms to detect convergence. The first and most obvious is that the optimization process will automatically be terminated after a user defined number of design cycles. The default limit is ten design cycles, and this will normally produce a high quality optimum, assuming a reasonable initial design was provided.

Beyond this, *GENESIS* considers both “Soft Convergence” and “Hard Convergence” tests. Soft convergence is defined as the case where no further progress can be made (i.e., the design variables do not change). Since the design variables did not change, it is considered unnecessary to perform an additional detailed analysis and repeat the process. Hard convergence occurs when two consecutive design cycles do not improve the optimum appreciably, even though significant changes in the design variables are occurring. In this case, since the design variables did change, a detailed analysis is required to insure the quality of the proposed design.

On contact analysis optimization problems, if soft or hard converge occurs on a design cycle where no full contact analysis was performed, the program will not stop and it will issue a warning message.

2.2.6 Matching Analysis Results

Sometimes it is desirable to tune a finite element model to match experimental data generated by tests of an actual structure. This is referred to here as matching analysis results. There are currently two methods available in *GENESIS* to do this: the Least Squares method and the Beta method.

In the least squares method, the objective function is the sum of the squared, normalized, differences between the actual responses and those calculated by the finite element analysis. This can be stated as;

$$F = \sum_{i=1}^{NR} [M_i(R_i - T_i)]^2 \quad (\text{Eq. 2-17})$$

where T_i is the target value of the response, R_i is the calculated value, NR is the number of responses to be matched and M_i is a multiplier calculated using the following expression:

$$\begin{aligned} M_i &= \frac{W_i}{\text{Max}(|T_i|, \text{RMATCH})} & \text{if}(W_i > 0.0) \\ M_i &= |W_i| & \text{if}(W_i < 0.0) \\ M_i &= \frac{1.0}{\text{Max}(|T_i|, \text{RMATCH})} & \text{if}(W_i = \text{blank}) \end{aligned} \quad (\text{Eq. 2-18})$$

in the above expressions, W_i is a weighting factor specified in **DMATCH2** (**DMATCH** sets all W_i to 1.0) and the **DOPT** parameter **RMATCH** is 0.01 by default.

Constraints on the other responses can also be included in the optimization problem formulation

In the Beta method, the objective function is the internal variable called BETA. An internal set of constraints is added to the original constraints to force the matching. The internal problem is;

$$\text{Minimize } \beta \quad (\text{Eq. 2-19})$$

$$\text{S.T. } -\beta \leq M_i(R_i - T_i) \leq \beta \quad i = 1, NR \quad (\text{Eq. 2-20})$$

In addition, all of the originally defined constraints are included, though they are not listed here.

The **DOPT** parameter **IMATCH** can be used to select the matching method. A value of 1 will select the Beta method and a value of 0 (Default) will select the Least Squares method.

2.2.7 Stress Ratio Method

The stress ratio method is an optimization method applicable to a special subset of structural optimization problems. It is used to resize members of a structure such that certain static stress constraints are exactly satisfied. In *GENESIS*, this method is typically used in conjunction with the standard method. When selected, the stress ratio method is used for the first **ISRMAX** design cycles and after that *GENESIS* will use the standard approximation method. **ISRMAX** is a **DOPT** parameter with a default value of 3. With the stress ratio method, *GENESIS* can be used to design areas of rods and/or thicknesses of shells or composites subject to stress and/or failure index constraints. In problems that contain other constraints, *GENESIS* will temporarily ignore them while using the stress ratio method. They will be considered when the approximation method gets activated.

Due to its limited scope, this method is not recommended for most problems. This method should only be used on the rare occasion when the standard method has problems resizing elements.

The advantage of the stress ratio method over the standard method is that it does not require the calculation of sensitivities or the solution of the approximate problem. This means that each design cycle can be faster.

It is interesting to note that the stress ratio method, when used to optimize a statically determinate structure that only has stress constraints, can converge in one design cycle. In non-statically determinate structures this is usually not true.

If this method is used alone, it may not converge to an optimal solution. The reason for this is that an optimal solution usually does not require that all members be fully stressed in at least one load case.

To update areas of rods, the stress ratio method uses the following equation:

$$\text{Area new} = \text{MAX}(\text{Area old} * \text{Stress} / \text{Stress limit}, \text{PMIN})$$

To update thickness of plates the stress ratio method uses the following equation:

$$\text{thickness new} = \text{MAX}(\text{thickness old} * \text{Stress} / \text{Stress limit}, \text{PMIN})$$

Where,

PMIN is the minimum property value as defined in DVPROP1.

The **DOPT** parameter, **ISMET**, is used to control the stress ratio resizing method. If **ISMET**=0, *GENESIS* will not use stress ratio (this is the default). If **ISMET** = 1 or 3, *GENESIS* will not change the design variables that reference elements that do not have stress ratio constraints. The difference between 1 and 3 is that the later case will not stop due to hard convergence until at least one cycle using the standard approximation method has been completed. Between 1 and 3, 3 is recommended for most cases. Option 1 should be picked over 3 only when analysis is very time consuming. If **ISMET**=2 or 4, *GENESIS* will attempt change the design variables associated to all stress ratio designable elements (e.g., shell elements without stress

constraints will go to their minimum gage). The difference between 2 and 4 is that the later case will not stop due to hard convergence until at least one cycle using the standard approximation method has been completed. Between 2 and 4, 4 is recommended for most cases. Option 2 should be pick over 4 only when analysis is very time consuming.

2.2.8 The STRDOT Optimizer

STRDOT is a new and experimental optimizer. STRDOT can be used to solve a special subset of structural optimization problems. It is used to resize members of a structure so that certain static stress constraints are satisfied. In *GENESIS*, this method is typically used in conjunction with the standard method. When selected, the STRDOT optimizer is used for the first **ISDMAX** design cycles and after that *GENESIS* will use the standard approximation methods along with the DOT or BIGDOT optimizer. ISDMAX is a **DOPT** parameter with a default of 0. The user must set the ISDMAX value or STRDOT will not be used for any design cycles.

The STRDOT optimizer is primarily controlled by the DOPT parameter **STRDOT**. A value of 1 will turn on this optimizer. With this value, *GENESIS* can be used to design thicknesses of shells or composites subject to stress and/or failure index constraints. In problems that contain other constraints, *GENESIS* will temporarily ignore them while using this optimizer. The ignored constraints will be considered when the approximation method gets activated.

The STRDOT parameter can also take values of -1 or 0. A value of -1 will cause *GENESIS* to only use the STRDOT optimizer. In this case all constraints that are not considered by STRDOT will not be used in optimization and might be violated. A value of 0, the default, will cause *GENESIS* to not use STRDOT and only standard optimizers will be used.

Due to its limited scope, this method is not recommended for most problems. This method should only be used when there is a large number of stress constraints and/or when the standard method has problems resizing elements.

The advantage of this optimizer over the standard ones is that it requires much less disk space and memory to use it.

In most cases, STRDOT can be used to replace the stress ratio method. STRDOT will most likely make the stress ratio method obsolete.

2.2.9 Discrete Optimization

Design variables (**DVAR**) can reference discrete sets (created by **DVSET** and **DVSET1** bulk data entries) to allow for discrete optimization.

Several **DOPT** parameter are available to help the user control the process. The first and most important parameter is **DSTART**. This parameter controls when the discrete optimization process starts. The default for **DSTART** is -1, which causes *GENESIS* to perform first continuous optimization (with either **DOT** or **BIGDOT**) and when that converges, perform discrete optimization using the **BIGDOT** or **DSCDOT** optimizer.

The second parameter available to the user is **DVINIT2**. This parameter controls the way *GENESIS* handles the initial values of the discrete design variables. If the value of **INIT** specified on the **DVAR** entry is not in the associated design discrete set, by default *GENESIS* stops with a fatal error. The **DVINIT2** parameter can change this behavior by forcing *GENESIS* to replace **INIT** with the closest value in the discrete set or to use **INIT** even if it does not belong to the discrete set.

Other **DOPT** parameter of interest are: **DDELP**, **DDELC**, **DDELL**, **DDELA**, **DDPMIN**, **DDCMIN**, **DDLMIN**, **DDAMIN**, **PENLTD**, **IPEN**, **PMULTD** and **NDISCR**. These parameters should be used only in case difficulties are encountered.

Further details on how to use discrete variables can be found in **Discrete Design Variables** (p. 261).

GENESIS provides three optimizers to perform discrete optimization. **DSCDOT**, **CMBDOT** and **BIGDOT**. **BIGDOT** has been used in *GENESIS* since the late nineties, **DSCDOT** since 2008 and **CMBDOT** since 2018. To use **CMBDOT**, the **DOPT** parameter **DSCDOT** must be set to 2. To use **DSCDOT**, the **DOPT** parameter **DSCDOT** must be set to 1. To use **BIGDOT**, the **DOPT** parameter **DSCDOT** must be set to 0. The current default for the **DSCDOT** **DOPT** parameter is 1.

The **CMBDOT** optimizer uses a **DOPT** parameter named **MAXDPG** to control the maximum number of discrete variables that are use by **CMBDOT** on a given instance of using the optimizer. When the number of discrete variables exceeds **MAXDPG**, *GENESIS* will call **CMBDOT** using additional instances of **CMBDOT** calls. It is recommended to not exceed a value of 10 for **MAXDPG** as larger values will require a very large number of function evaluations, causing the process to slow down. The number of function evaluations required grows exponentially with **MAXDPG**.

The **CMBDOT** optimizer may provide better results than **DSCDOT**, but will usually require significantly more computation time. To speed up **CMBDOT**, use a lower value of **MAXDPG**.

For problems with relatively few design variables, **CMBDOT** is recommended, while for problems with larger numbers of design variables, **DSCDOT** is recommended.

2.2.10 Nonlinear Contact Analysis Optimization

Nonlinear contact analysis requires the solver to iterate many times to get accurate answers. This iteration causes the analysis to be much slower than simple linear static analysis. When performing optimization with a nonlinear analysis, it is possible to use a technique termed “simultaneous analysis and optimization”. Since the optimization is going to perform many design cycles, it might be not necessary to have a precise nonlinear answer in each design cycle. In this class of problems, the user might want to limit the number of full contact analyses. The user might also want to limit the number of full contact analyses when the solver time become too large.

The DOPT parameter **CNTDC** allows the user to control the frequency at which full contact analysis is performed. The default, 1, causes the program to perform full contact analysis in every design cycle. A value n , greater than one, causes the program to perform full contact analysis every n -th design cycle. The program, however, will always perform full contact analysis on the first and last design cycles. For the design cycles that are not performing full contact analysis, the DOPT parameter **CNTIT** sets a reduced maximum number of contact analysis iterations.

The following figure shows an example for $CNTDC=4$ that assumes optimization finished in design cycle 10. The figure shows that $CNTDC=4$ causes the program to perform full contact analysis in design cycles 4 and 8. Design cycles 0 and 10 are performed by the program too which provides for good starting and final designs.

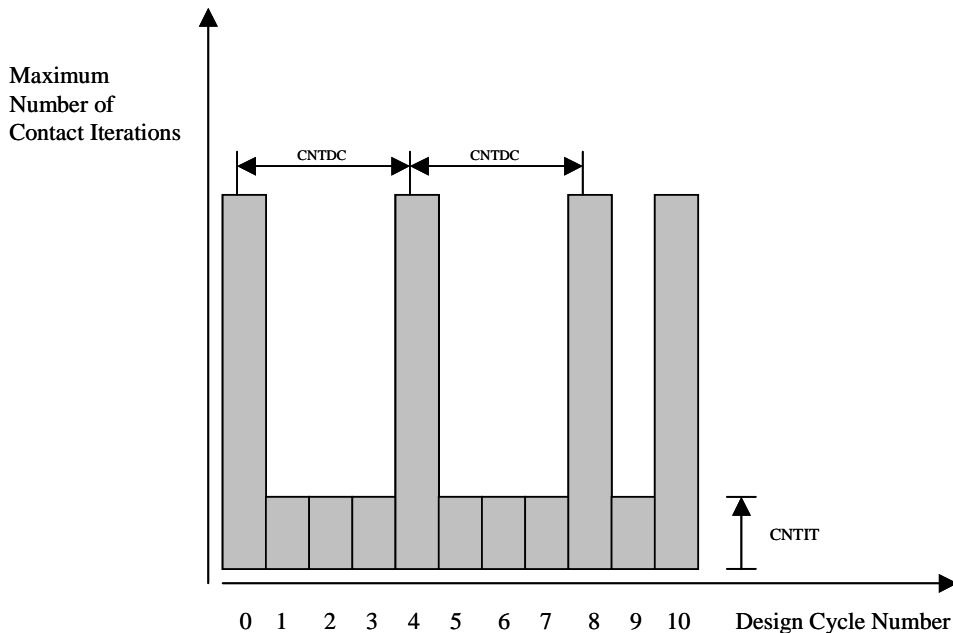


Figure 2-1

If soft or hard converge occurs on a design cycle where no full contact analysis was performed, the program will not stop, but instead trigger a full contact analysis on the next design cycle.

2.3 Shape, Sizing Topography and Topometry Design Bulk Data

This section summarizes the shape, sizing, topography and topometry design input data to *GENESIS*. The design data is placed with the analysis data between the BEGIN BULK and the ENDDATA commands.

Design Variables

NAME	INFORMATION
DVAR	Design variable definition
DVSET	Design variable discrete value list
DVSET1	Design variable discrete value list
DLINK	Linear linking between design variables

Design Variable To Property And Grid Relationships

NAME	INFORMATION
DVPROP1	Linear relationship between design variables and properties
DVPROP2	Synthetic relationship between design variables and properties
DVPROP3	Built in design element library relationships for beams and plates
DVPROP4	Linear relationship between design variables and composite thicknesses and /or angles
DSPLIT	Defines a topometry region, symmetry conditions and coarse levels
DOMAIN	Define a set of grids (DOMAIN Element) to be used with DVGRIDC data to create basis or perturbation vectors.
DSHAPE	Defines data for DVGRID/DVGRIDC sets
DVGRID	Linear relationship between design variables and grid point locations
DVGRIDC	Linear relationship between design variables and corner grids, or middle edge grids, of DOMAIN elements.
DTGRID	Defines data needed to generate design data for topography regions.
DEQATN	Equation referenced by DVPROP2
DTABLE	Tabled constants for equations
DLIB	Library element information defined by the user

Responses

NAME	INFORMATION
DRESP1	Identifies direct response to be used as objective function or constraint or entry to DRESP2
DRESP2	Synthetic relationship between design variables, grid locations, DRESP1 and DRESPG quantities
DRESP3	Relationship defined by an external analysis program or by a built-in equation that is a function of grid point locations, design variable values, DRESP1 and DRESPG quantities
DRESPG	Defines a geometric response to be used in optimization
DRESPU	Identifies responses generated by an external analysis program
DEQATN	Equation referenced by DRESP2
DTABLE	Tabled constants for equations
DSHIFT	Frequency response shifting parameters

Objective Function

NAME	INFORMATION
DOBJ	Identifies a DRESP1, DRESP2, DRESP3, DRESPG or DRESPU entry as the design objective
DINDEX	Identifies one or more of DRESP1, DRESP2, DRESP3, DRESPG or DRESPU entries and corresponding weighting factors to form a design objective.

Matching Analysis Results

NAME	INFORMATION
DMATCH	Identifies a DRESP1, DRESP2, DRESP3, DRESPG or DRESPU entry and values that they are to match
DMATCH2	Identifies a DRESP1, DRESP2, DRESP3, DRESPG or DRESPU entry and values that they are to match, along with weighting factors

Constraints

NAME	INFORMATION
DCONS	Identifies a DRESP1, DRESP2, DRESP3 or DRESPG entry as a constraint and provides bounds
DCONS2	Identifies a DRESP1, DRESP2, DRESP3 or DRESPG entry as a constraint and provides bounds as scale factors of the response's initial analysis value
DSCREEN	Constraint screening information for sensitivity analysis and approximate optimization

Design Variable Selection

NAME	INFORMATION
DSELECT	Defines information used to add constraints intended to force the selection of a subset of design variables.

Parameters

NAME	INFORMATION
DOPT	Defines various optimization parameters

Relationships Among Shape and Sizing Design Data

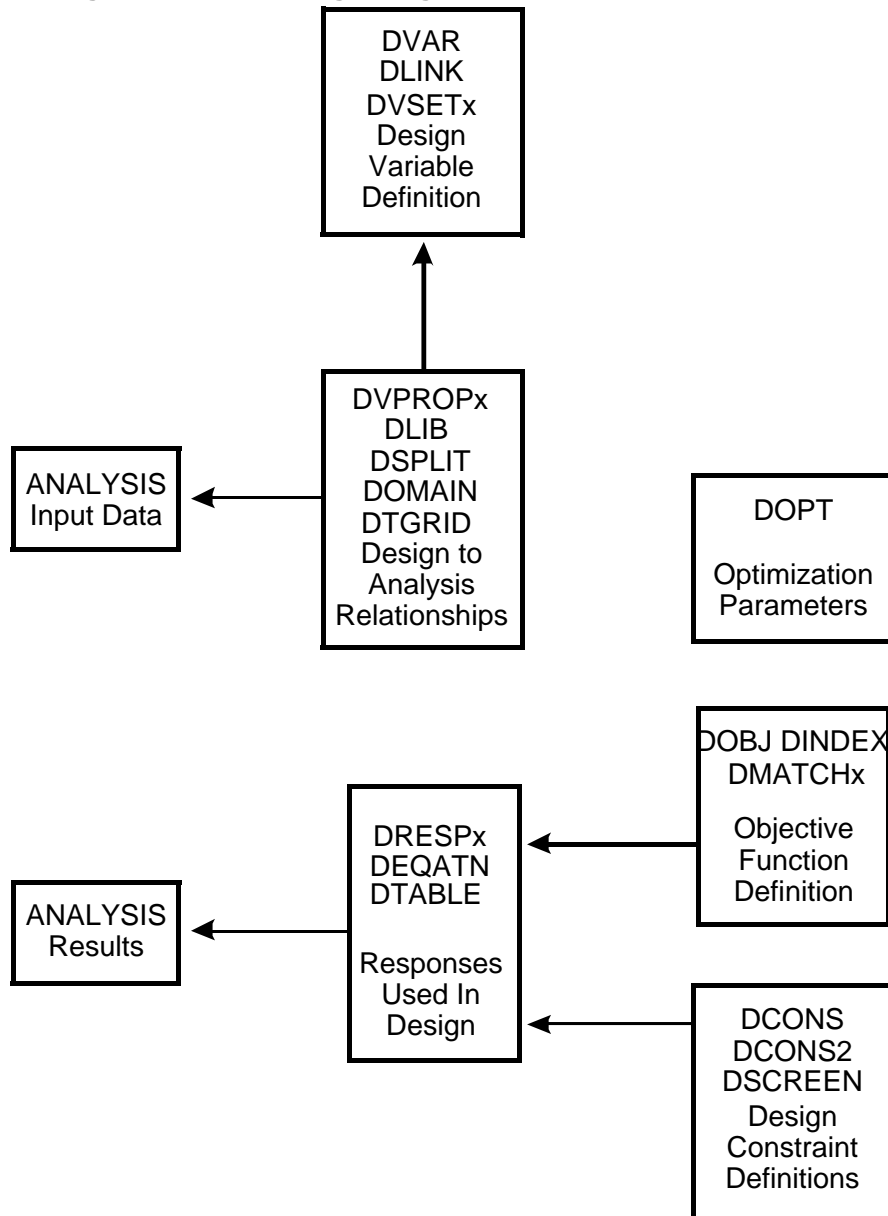


Figure 2-2

2.4 Topology Design Bulk Data

This section summarizes the topology design data. The design data is placed with the analysis data between the BEGIN BULK and ENDDATA statements.

Topology Regions

NAME	INFORMATION
TPROP	Selects a property whose elements are to be topologically designed.
TPROPC	Selects a property whose elements are to be topologically copied from another topology designed region.

Responses

NAME	INFORMATION
TRESP1	Identifies direct responses to be used as the objective function or constraints.
TRESP2	Synthetic relationship between extra variables and TRESP1 quantities
TRESP3	Relationship defined by an external analysis program or by a built-in equation that is a function of extra variable values and TRESP1 quantities
DEQATN	Equation referenced by TRESP2
DTABLE	Tabled constants for equations
DSHIFT	Frequency response shifting parameters

Objective Function

NAME	INFORMATION
TOBJ	Identifies a TRESP1, TRESP2 or TRESP3 as the design objective.
TINDEX	Identifies TRESP1, TRESP2 or TRESP3 entries and weighting factors used to form the compliance index objective function.

Constraints

NAME	INFORMATION
TCONS	Identifies a TRESP1, TRESP2 or TRESP3 as a constraint and provides bounds.
TCONS2	Identifies a TRESP1, TRESP2 or TRESP3 as a constraint and provides bounds as scale factors of the response's initial analysis value.
DSCREEN	Constraint screening information.

Fabrication Constraints

NAME	INFORMATION
TSYM1	Identifies a symmetry plane based on grids. Also used to defined fabriacation constraints.
TSYM2	Identifies a symmetry plane based on points. Also used to defined fabriacation constraints.
TSYM3	Identifies a symmetry plane based on a coordinate system. Also used to defined fabriacation constraints.

Extra Variables

NAME	INFORMATION
TVAR	Topology extra variable definition

Design Variable Selection

NAME	INFORMATION
TSELECT	Defines information used to add constraints intended to force the selection of a subset of topology extra variables (TVAR) or internal topology variables.

Parameters

NAME	INFORMATION
DOPT	Defines various optimization parameters.

The chart below shows the basic relationships among the data statements for design using *GENESIS* for topology optimization. It includes all topology design commands that may be included in the input data file. The chart also includes the most relevant analysis data that are referenced by the topology design data.

Bulk Data

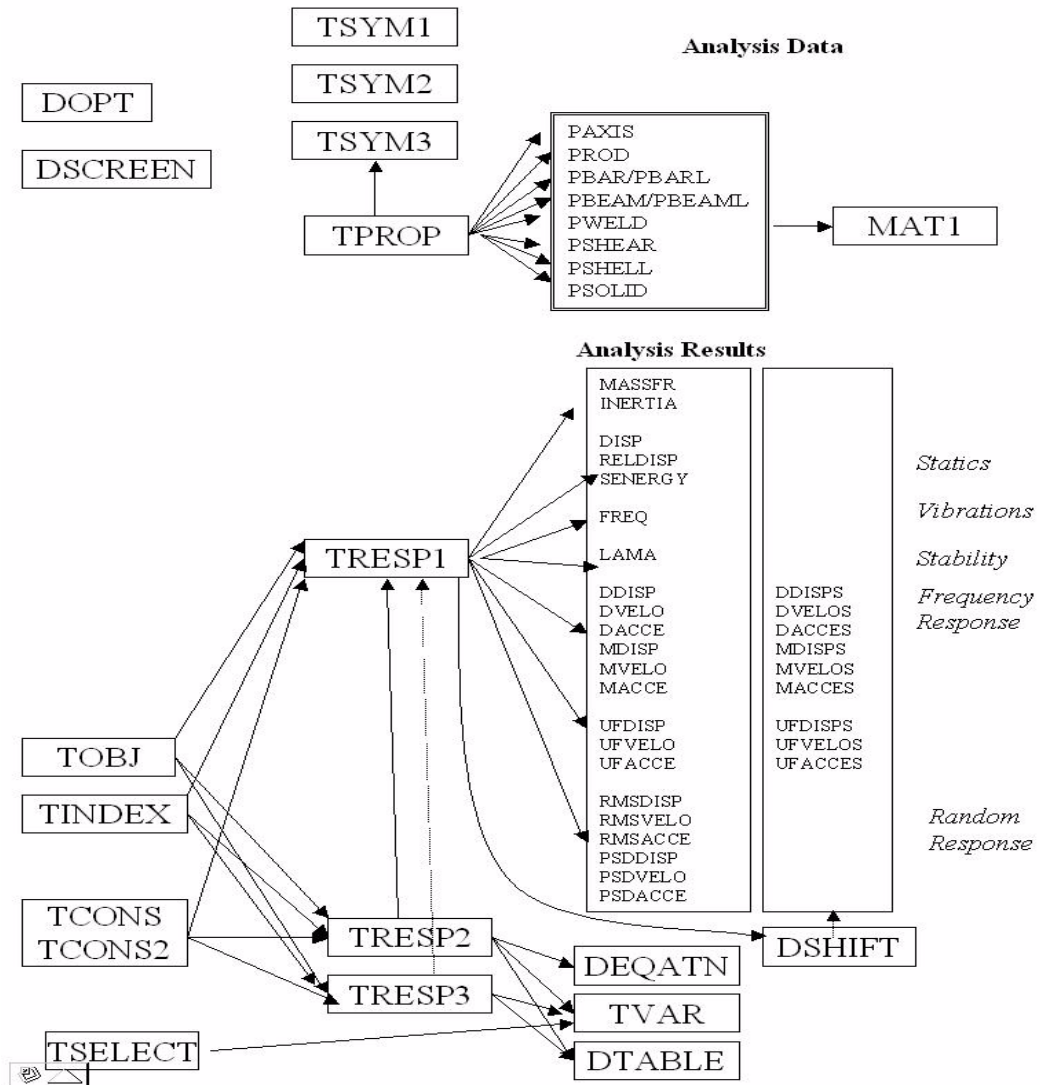


Figure 2-3

CHAPTER 3

Shape and Sizing Design Capabilities

3

- Basic Sizing Design Capabilities
- Basic Shape Design Capabilities
- Advanced Design Capabilities
- Advanced Shape Design Capabilities
- Reliability Based Optimization
- Freeform Optimization
- Topography Optimization
- Topometry Optimization

3.1 Basic Sizing Design Capabilities

The analysis engineer deals with element cross sectional dimensions (height, thickness, etc.), section properties (area, bending moment of inertia, etc.), structural dimensions (length, span, etc.), and responses (displacements, stresses, frequencies). The cross sectional dimensions, section properties, and grid point locations define the analysis model. The design engineer deals with design variables that, in general, have some sort of nonlinear relationship with the description of the analysis model, and constraints on the responses generated by the analysis model. The collection of design variables, constraints, and the objective function define the design model. The basic design capabilities are used to relate the analysis model to a basic design model.

Sizing Designable Elements

Following is a list of all elements designable with sizing optimization:

SIZING DESIGNABLE ELEMENTS	
Element	Property
CELAS1	PELAS
CVECTOR	PVECTOR
CBUSH	PBUSH
CMASS1	PMASS
CONM3	PCONM3
CDAMP	PDAMP
CVISC	PVISC
CROD	PROD
CBAR	PBAR
CHBDY	PHBDY
CSHEAR	PSHEAR
CTRIA3	PSHELL/PCOMP/PCOMPG
CQUAD4	PSHELL/PCOMP/PCOMPG
CTRIA6	PSHELL/PCOMP/PCOMPG
CQUAD8	PSHELL/PCOMP/PCOMPG
CTRIAX6	PAXIS
K2UU1	PK2UU
M2UU1	PM2UU

3.1.1 Design Variable Definition (DVAR)

Each design variable has an initial value and lower and upper bounds. The initial value usually comes from the engineers preliminary design model. The lower and upper bounds are used to keep the design from becoming unreasonable. For example the wheelbase of a car must be less than the width of the road. Lower bounds are usually the results of manufacturing considerations. For example it is difficult to manufacture rod elements with a diameter of 0.001 mm. Each design variable has an ID number by which it is referenced and an optional label which is provided to help the user quickly identify it in the program output.

The basic format for the design variable data in *GENESIS* is:

1	2	3	4	5	6	7	8	9	10
DVAR	ID	LABEL	INIT	LB	UB				

3.1.2 Linear Design Variable to Element Property Relationship (DVPROP1)

The DVPROP1 input data is used to construct linear relationship between one or more design variables and a finite element property listed on the PAXIS, PROD, PBAR, PBARL, PBUSH, PSHELL, PCOMP, PCOMPG, PELAS, PELASH, PK2UU, PM2UU, PVECTOR, PSHEAR, PCONM3, PHBDY, PDAMP, PMASS or PVISC data. The design variables are specified by their ID numbers. The property is specified by its property input data ID (PID) and a code, called the FID, which points to a specific property on the input data. These codes are found with the DVPROP1 input data described in **DVPROP1** (p. 603) For example, the code for the area of a ROD is 1 and the code for the thickness of a QUAD4 element is 2.

The basic format of the DVPROP1 input data is:

	1	2	3	4	5	6	7	8	9	10
DVPROP1	ID	PID	FID	C0						
+	DVID1	C1	DVID2	C2						

The ID is the unique DVPROP1 identifier. The PID and FID define the property that is being controlled by the design variables DVIDi and C0, C1, C2, ... which are used in the relationship

$$\text{Property} = C_0 + \sum_i C_i \cdot \text{DVID}_i \quad (\text{Eq. 3-1})$$

The most basic use of the DVPROP1 data is to form a direct one to one relationship between a design variable and some element property. For example, to link the areas of the ROD elements associated with PROD 6 with the value of design variable 10 the data would look like:

	1	2	3	4	5	6	7	8	9	10
DVPROP1	14	6	1							
+	10									

where 14 is the DVPROP1 ID. Note that the default value for C0 is 0 and for C1 is 1.0. Therefore:

$$\text{Area}_6 = \text{DV}_{10} \quad (\text{Eq. 3-2})$$

A more complicated example of the use of the DVPROP1 data would be to assign the thickness of plate/shell elements that reference PSHELL 20 to be 2.0 plus the average values of design variables 15 and 16. In this case the data would appear as:

	1	2	3	4	5	6	7	8	9	10
DVPROP1	14	20	2	2.0						

Shape and Sizing Design Capabilities

1	2	3	4	5	6	7	8	9	10
+	16	0.5	15	0.5					

Note that, while there is no limit to the number of design variables that can be referenced by the DVPROP1 data, only a single property can be specified.

Shape and Sizing Design Capabilities

The DVPROP1 data can also be used to create sizing Basis Designs (Basis Vectors) for optimization. For example, consider the design of a ROD assembled with 5 CROD elements. Assume that we have the following three candidate designs and wish to find the best linear combination of these.

BASIS 1	BASIS 2	BASIS 3
A1=1.0	A1=1.0	A1=1.0
A2=1.0	A2=0.8	A2=0.8
A3=1.0	A3=0.6	A3=0.6
A4=1.0	A4=0.4	A4=0.8
A5=1.0	A5=0.2	A5=1.0

These shapes are shown in **Figure 3-1** below.

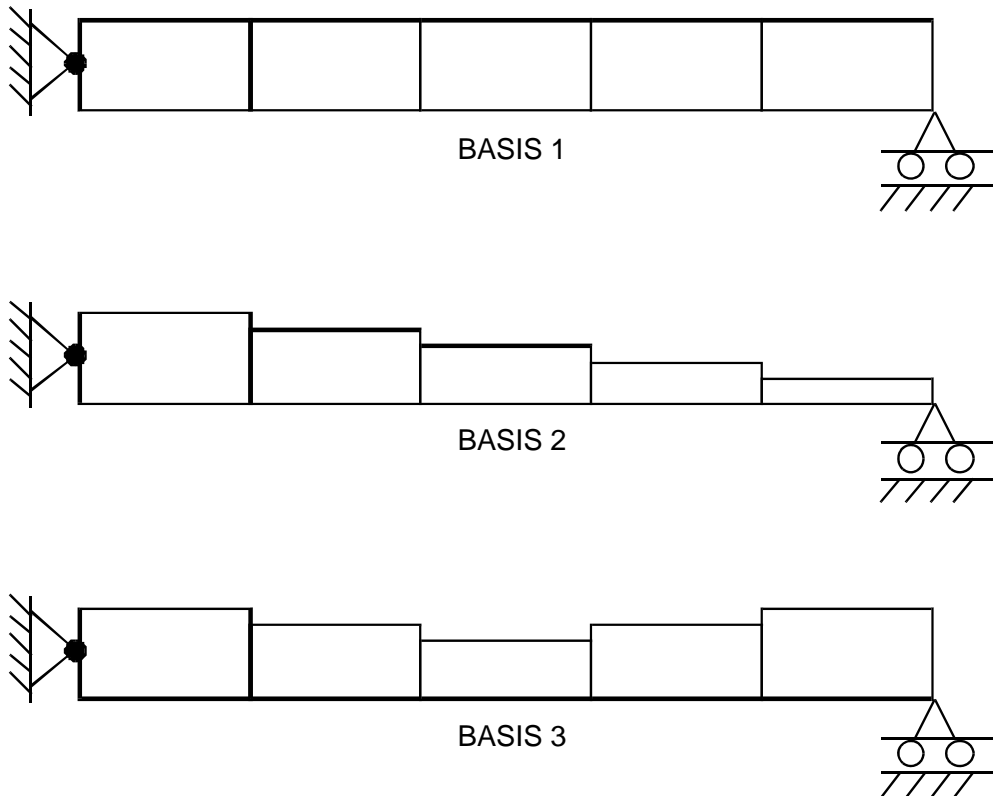


Figure 3-1 Basic Design

An optimum design can be found as a linear combination of these three designs, so we can define the member areas as:

$$\begin{Bmatrix} A1 \\ A2 \\ A3 \\ A4 \\ A5 \end{Bmatrix} = DV1 \begin{Bmatrix} 1.0 \\ 1.0 \\ 1.0 \\ 1.0 \\ 1.0 \end{Bmatrix} + DV2 \begin{Bmatrix} 1.0 \\ 0.8 \\ 0.6 \\ 0.4 \\ 0.2 \end{Bmatrix} + DV3 \begin{Bmatrix} 1.0 \\ 0.8 \\ 0.6 \\ 0.8 \\ 1.0 \end{Bmatrix} \quad (\text{Eq. 3-3})$$

To solve this problem, three DVAR data statements are required, corresponding to the three design variables. Five DVPROP1 data statements are required to control the member properties.

Thus, the appropriate data statements would be:

	1	2	3	4	5	6	7	8	9	10
DVAR	1	DV1	0.333	-10.0	10.0					
DVAR	2	DV2	0.333	-10.0	10.0					
DVAR	3	DV3	0.333	-10.0	10.0					
DVPROP1	100	1	1		0.01					
+	1	1.0	2	1.0	3	1.0				
DVPROP1	200	2	1		0.01					
+	1	1.0	2	0.8	3	0.8				
DVPROP1	300	3	1		0.01					
+	1	1.0	2	0.6	3	0.6				
DVPROP1	400	4	1		0.01					
+	1	1.0	2	0.4	3	0.8				
DVPROP1	500	5	1		0.01					
+	1	1.0	2	0.2	3	1.0				

In this example, the initial design is the average of the three candidate designs and each design variable has a lower bound of -10.0 and an upper bound of 10.0. Note that the design variables can have negative values. However, the properties must not be negative. Therefore, a value of PMIN=0.01 is used on the DVPROP1 data to insure that the cross-sectional areas are positive. The PMIN value is placed in the sixth column of DVPROP1 data.

Shape and Sizing Design Capabilities

In many structural design problems the finite element properties are not linear functions of the design variables. For example the bending moment of inertia of a beam is a nonlinear function of the height of the beam and the bending stiffness of a plate element is a nonlinear function of the thickness. In these cases the nonlinear design variable to property relationship (DVPROP2) data or design element library data (DVPROP3) must be used. See [DVPROP3](#) (p. 621) for more detailed information on this data.

3.1.3 Design Variable Linking (DLINK)

The DLINK input data is used to link a dependent design variable to independent design variables by a linear relationship. This relationship is defined as:

$$DV^{\text{dep}} = C_0 + C_{\text{mult}} \sum_i C_i * DV_i^{\text{indep}} \quad (\text{Eq. 3-4})$$

The format of the DLINK data is:

1	2	3	4	5	6	7	8	9	10
DLINK	DVID	C0	CMULT	DV1	C1	DV2	C2	DV3	C3
+	DV4	C4	DV5	C5	...				Blank

Note that the continuation line is not needed if only one, two or three independent design variables are referenced. The simplest linking is one to one. For example dependent design variable 6 is equal to independent design variable 9. In this case the input data is:

1	2	3	4	5	6	7	8	9	10
DLINK	6	0.0	1.0	9	1.0				

or

1	2	3	4	5	6	7	8	9	10
DLINK	6			9	1.0				

Note that the default values for C0 and CMULT are 0.0 and 1.0 respectively. There are no default values for the C_i .

Many times designers develop tapered structures. One approach to the design of these types of structures is to have independent design variables control each end, while dependent design variables control the interior. For example consider a structure composed of five equal length ROD elements whose areas are controlled by design variables 1 through 5. The equations for the areas of RODs 2, 3, and, 4 are

$$A_2 = \frac{3}{4}A_1 + \frac{1}{4}A_5 \quad (\text{Eq. 3-5})$$

$$A_3 = \frac{1}{2}A_1 + \frac{1}{2}A_5 \quad (\text{Eq. 3-6})$$

$$A_4 = \frac{1}{4}A_1 + \frac{3}{4}A_5 \quad (\text{Eq. 3-7})$$

Shape and Sizing Design Capabilities

DLINK data for this problem is:

	1	2	3	4	5	6	7	8	9	10
DLINK		2		0.25	1	3.0	5	1.0		
DLINK		3		0.5	1	1.0	5	1.0		
DLINK		4		0.25	1	1.0	5	3.0		

As a final example, in many symmetric structures one design variable is the negative of another. In this case the DLINK data is:

	1	2	3	4	5	6	7	8	9	10
DLINK		6			9	-1.0				

3.1.4 Structural Responses (DRESP1)

The structural responses of the analysis model are used to construct constraint or objective functions in the design model. The DRESP1 input data is used to tag the important structural responses for the design model. These responses can be system responses: mass, volume, moment of inertia, strain energy and frequency; grid point responses: displacements, velocities, accelerations, grid stresses (for TRIAX6, HEXA, HEX20, PENTA, TETRA and PYRA elements), temperatures; or element responses: stress, strain, and force. Element responses can be referenced element by element or property by property. When referenced by property all of the elements that reference each property are tagged.

The basic format or the DRESP1 data is:

1	2	3	4	5	6	7	8	9	10
DRESP1	ID	LABEL	RTYPE	PTYPE		ATTA	ATT1	ATT2	ATT3
+	ATT4	...							

Each DRESP1 data has a unique ID, which is required, and an optional LABEL that can be used to identify it in the program output. The response type (RTYPE) must be MASS, VOLUME, INERTIA, FREQ, RFREQ, LAMA, DISP, SPCF, EVECT, REVECT, DDISP, DDISPS, MDISP, MDISPS, DVELO, DVELOs, MVELO, MVELOs, DACCE, DACCES, MACCE, MACCES, RMSDISP, RMSVELO, RMSACCE, PSDDISP, PSDVELO, PSDACCE, STRESS, DSTRESS, DSTRS, GSTRESS, DGSTRESS, CSTRESS, FINDEX, CTHICK, LTHICK, STRAIN, DSTRAIN, DSTNS, CSTRAIN, SENERGY, VMINDEX, CDISP, CPRESS, TEMP, FORCE, DFORCE, DFORCES, UFDISP, UFDISPS, UFVELO, UFVELOs, UFACCE, UFACCES, ERP or ERPS. If the response type is SENERGY or VMINDEX, then the rest of the data fields are not used. If the response type is TEMP, then the ATTI are the grid points. If the response type is MASS or VOLUME and the rest of the data fields are blank, then the total system mass or volume is referenced. If the response type is MASS or VOLUME and the PTYPE is PROP or MAT, then the ATTI are the property or material ID's. If the response type is FREQ or RFREQ, then the PTYPE data will be the mode number. If the response type is EVECT or REVECT, then the PTYPE data will be the mode number, the attribute data (ATTA) will be the degree or degrees of freedom, and the ATTi will be the grid point. If the response type is LAMA, then the PTYPE is the buckling mode number. If the response type is DISP then the attribute data will be the referenced degree or degrees of freedom and the ATTi will be the grid points. If the response type is GSTRESS, the attribute data (ATTA) will be the stress item code and the ATTi will be the grid points.

If the response type is INERTIA, the attribute data is the inertia item code.

For element responses, if the DRESP1 data references elements directly, then PTYPE should be “ELEM”. If the DRESP1 data references elements by referencing their property data (the default) then PTYPE should be PROP, the property type (PROD, PSHELL, PELAS, etc.), or blank. The ATTi are either the element or property ID’s. The ATTA data is the stress, strain, or force item code. These element response codes are found with the **DRESP1** (p. 522) of Volume II.

For dynamic responses real, imaginary, magnitude and phase components can be referenced by DRESP1 data. Note that the magnitude of a response is always positive. Care should be taken in constraining the phase of a response since it is limited to the range of 0-360°. Note also that if a phase is constrained to be less than 2° and it becomes -2° = 358°, the constraint will be violated.

The RTYPE, PTYPE, ATTA, and ATTi data are summarized in the table below.

RESPONSE	RTYPE	PTYPE	ATTA	ATTi
Static Displacement	DISP	Blank	Displacement component code. See remark 5.	Grid or Spoint ID or “GSET”, “THRU” or “ALL”
Relative Displacement	RELDISP	Blank	Displacement component code.	Grid or Spoint
Reaction Force	SPCF	Blank	Reaction Force component code (1-7)	Grid or Spoint
Static Element Stress	STRESS	Blank or PROP, PAXIS, PBAR, PBEAM, PBUSH, PROD, PSHELL, PSOLID, PELAS, PSHEAR, ELEM	Stress item code	Property entry (PID) or Element ID (EID) or “ESET”
Dynamic Element Stress	DSTRESS	Blank or PROP, PAXIS, PBAR, PBEAM, PBUSH, PROD, PSHELL, PSOLID, PELAS, PSHEAR, ELEM	Stress item code	Property entry (PID) or Element ID (EID) or “ESET”
Shifted Dynamic Element Stress	DSTRS	Blank or PROP, PAXIS, PBAR, PBEAM, PBUSH, PROD, PSHELL, PSOLID, PELAS, PSHEAR, ELEM	Stress Magnitude item code	ATT1: DSHIFT ID. ATTi (i>1): Property entry (PID) or Element ID (EID)

RESPONSE	RTYPE	PTYPE	ATTA	ATTi
Static Grid Stress	GSTRESS	Blank	Stress item code	Grid ID or "GSET", "THRU" or "ALL"
Dynamic Grid Stress	DGSTRESS	Blank	Stress item code	Grid ID or "GSET", "THRU" or "ALL"
Composite Layer Stress	CSTRESS	Blank, PCOMP or ELEM	Stress item code	ATT1: layer number. ATTi (i>1): Property entry (PID) or Element ID (EID)
Composite Element Failure Index	FINDEX	Blank, PCOMP or ELEM	Index item code	Property entry (PID) or Element ID (EID) or "ESET"
Static Element Strain	STRAIN	Blank or PROP, PAXIS, PBUSH, PSHELL, PSOLID, ELEM	Strain item code	Property entry (PID) or Element ID (EID) or "ESET"
Dynamic Element Strain	DSTRAIN	Blank or PROP, PAXIS, PBUSH, PSHELL, PSOLID, ELEM	Strain item code	Property entry (PID) or Element ID (EID) or "ESET"
Shifted Dynamic Element Strain	DSTNS	Blank or PROP, PAXIS, PBUSH, PSHELL, PSOLID, ELEM	Strain Magnitude item code	ATT1: DSHIFT ID. ATTi (i>1): Property entry (PID) or Element ID (EID)

RESPONSE	RTYPE	PTYPE	ATTA	ATTi
Composite Layer Strain	CSTRAIN	Blank, PCOMP or ELEM	Strain item code	ATT1: layer number. ATTi (i>1): Property entry (PID) or Element ID (EID)
Static Element Force	FORCE	Blank or PROP, PBAR, PBEAM, PBUSH, PSHELL, PELAS, PSHEAR, PROD, PVECTOR, ELEM	Force item code	Property entry (PID) or Element ID (EID) or "ESET"
Dynamic Element Force	DFORCE	Blank or PROP, PBAR, PBEAM, PBUSH, PCOMP, PSHELL, PELAS, PSHEAR, PROD, PDAMP, PVECTOR, PVISC, ELEM	Force item code	Property entry (PID) or Element ID (EID) or "ESET"
Shifted Dynamic Element Force	DFORCES	Blank or PROP, PBAR, PBEAM, PBUSH, PCOMP, PSHELL, PELAS, PSHEAR, PROD, PDAMP, PVECTOR, PVISC, ELEM	Force Magnitude item code	ATT1: DSHIFT ID. ATTi (i>1): Property entry (PID) or Element ID (EID)
Composite Total Thickness	CTHICK	Blank, PCOMP or ELEM	1	Property entry (PID) or Element ID (EID)
Composite Layer Thickness	LTHICK	Blank, PCOMP or ELEM	Thickness item code: 1 = layer thickness 2 = layer thickness / total thickness	ATT1: layer number. ATTi (i>1): Property entry (PID) or Element ID (EID)
Grid Contact Clearance	CDISP	Blank	Contact component code: 1	Grid ID or "GSET", "THRU" or "ALL"

RESPONSE	RTYPE	PTYPE	ATTA	ATTi
Grid Contact Pressure or Grid glue connection pressure	CPRESS	Blank	Contact component code: 1	Grid ID or "GSET", "THRU" or "ALL"
Natural Frequency	FREQ, RFREQ	Mode number	Blank	Blank
Eigenvector Component	EVECT, REVECT	Mode number	Component code	Grid or Spoint ID or "GSET", "THRU" or "ALL"
Buckling Load Factor	LAMA	Mode number	Blank	Blank
Mass	MASS	Blank or MAT or PROP	Blank	Blank or Property entry (PID) or Material entry (MID)
Volume	VOLUME	Blank or MAT or PROP	Blank	Blank or Property entry (PID) or Material entry (MID)
Static Strain Energy	SENERGY	Blank or PROP or PAXIS or PBAR or PBARL or PBEAM or PBEAML or PBUSH or PROD or PSHELL or PSHEAR or PCOMP or PCOMPG or PSOLID or PELAS or PVECTOR or PBUSH or PBUSHT or PGARP or PWELD or PK2UU	Blank	Blank or Property entry (PID)
Von Mises Stress Index	VMINDEX	Blank	Blank	Blank
Temperature	TEMP	Blank	Blank	GRID or SPOINT ID or "GSET", "THRU" or "ALL"
Heat Transfer Compliance	HTC	Blank	Blank	Blank

RESPONSE	RTYPE	PTYPE	ATTA	ATTi
Dynamic Displacement, Velocity, Acceleration (From Direct Loadcases)	DDISP, DVELO, DACCE	Blank	Component code (1-24)	Grid or Spoint ID or "GSET"
Shifted Dynamic Displacement, Velocity, Acceleration (From Direct Loadcases)	DDISPS, DVELO, DACCES	Blank	Component code;(1-6)	ATT1: DSHIFT ID. ATTi (i>1): Grid or Spoint ID
Dynamic Displacement, Velocity, Acceleration (From Modal Loadcases)	MDISP, MVELO, MACCE	Blank	Component code (1-24)	Grid or Spoint ID or "GSET"
Shifted Dynamic Displacement, Velocity, Acceleration (From Modal Loadcases)	MDISPS, MVELO, MACCES	Blank	Component code: (1-6)	ATT1: DSHIFT ID. ATTi (i>1): Grid or Spoint ID
Random Power Spectral Density Dynamic Displacement, Velocity, Acceleration (From Direct or Modal Loadcases)	PSDDISP, PSDVELO, PSDACCE	Blank	Component code.	Grid or Spoint ID
Shifted Random Power Spectral Density Dynamic Displacement, Velocity, Acceleration (From Direct or Modal Loadcases)	PSDDDS, PSDVS, PSDAS	Blank	Component code: (1-6)	ATT1: DSHIFT ID. ATTi (i>1): Grid or Spoint ID

RESPONSE	RTYPE	PTYPE	ATTA	ATTi
Random Dynamic Displacement, Velocity, Acceleration (From Random Loadcases)	RMSDISP, RMSVELO, RMSACCE	Blank	Component code (1-6)	Grid or Spoint ID
User Function of Dynamic Displacement, Velocity, Acceleration	UFDISP, UFVELO, UFACCE	Blank	Field Point item code 1: Magnitude 2: Phase 3: Real 4: Imaginary	Field Point ID (Defined in UFDATA file)
Shifted User Function of Dynamic Displacement, Velocity, Acceleration (From Direct or Modal Loadcases)	UFDISPS, UFVELOS, UFACCES	Blank	Component code 1: Magnitude	ATT1: DSHIFT ID. ATTi (i>1): Field Point ID (Defined in UFDATA file)
Equivalent Radiated Power	ERP	Blank	Item code 1: Standard scale 2: Decibel Scale	Element set ID (Specified in ERPPNL entry)
Shifted Equivalent Radiated Power	ERPS	Blank	Component code 1: Standard scale 2: Decibel Scale	ATT1: DSHIFT ID. Element set ID (Specified in ERPPNL entry)
System Inertia Matrix Component	INERTIA	Blank	Inertia item code	Blank

The following examples show the preparation of the DRESP1 input data for different response types.

To tag the system volume, the data entry is:

1	2	3	4	5	6	7	8	9	10
DRESP1	10		VOLUME						

Shape and Sizing Design Capabilities

Note that the DRESP1 ID is 10, the optional label is not used, and the ATTA and ATTi data are blank. To tag the mass of material 3, the data entry is;

1	2	3	4	5	6	7	8	9	10
DRESP1	15		MASS	MAT			3		

To tag the system moment of inertia, I_{zz} , at the center of gravity, the data entry is:

1	2	3	4	5	6	7	8	9	10
DRESP1	18		INERTIA			3			

To tag the third system frequency the data entry is:

1	2	3	4	5	6	7	8	9	10
DRESP1	20	MODE3	FREQ	3					

Note that the label is “MODE3” and the ATTA and ATTi data are left blank.

To tag the first buckling load factor the data entry is:

1	2	3	4	5	6	7	8	9	10
DRESP1	21	LAMA1	LAMA	1					

Note that the label is “LAMA1” and the ATTA and ATTi data are left blank.

To tag the u eigenvector component of mode 6 of grids 2, 3, 5 and 7, the data entry is:

1	2	3	4	5	6	7	8	9	10
DRESP1	25		EVECT	6		1	2	3	7
+	5								

To tag the u displacement of grids 2, 3, 5, and 7 the data entry is:

1	2	3	4	5	6	7	8	9	10
DRESP1	30		DISP			1	2	3	7
+	5								

Note that the PTYPE data is blank and that the grid point ID's do not have to be in any particular order.

Now consider the axial force at end A of ROD elements 1 and 2. The force item code for end A of a ROD element is 1. Therefore, the data is:

1	2	3	4	5	6	7	8	9	10
DRESP1	40	RODFA	FORCE	ELEM		1	1	2	

Note that the PTYPE is “ELEM” to specify that the ATTi are element ID's and not property ID's. The element type does not have to be specified since all element ID's must be unique.

Suppose we want to flag the Von Mises stresses on the top and bottom surfaces of all the plate/shell elements associated PSHELL ID's 100 and 200. The two DRESP1 data would be:

	1	2	3	4	5	6	7	8	9	10
DRESP1	10	BOTTO M	STRESS				2	100	200	
DRESP1	11	TOP	STRESS	PROP			9	100	200	

Note that the PTYPE is the default value in DRESP1 10 and "PROP" in DRESP1 11. Both are the same since the default is "PROP". Also note that the stress item codes for the Von Mises stress on the bottom and top surfaces of plate/shell elements are 2 and 9 respectively.

Shape and Sizing Design Capabilities

For grid point stresses (GSTRESS) and temperatures (TEMP), the keyword “THRU” can be used in the ATTi list. For example, to tag the maximum shear stress at grids 1, 3, 4, 5, 10, 100, 101, 102 and 103, the DRESP1 data could be

1	2	3	4	5	6	7	8	9	10
DRESP1	10		GSTRESS			9	1	3	THRU
+	5	10	100	THRU	103				

The keyword “THRU” can be used many times in the ATTi list and anywhere in the list. The only restriction is that the grid ID after the “THRU” be greater than the grid ID before the “THRU.”

To specify all grid points for grid point stresses or temperatures, the keyword “ALL” is used and the other ATTi must be blank. For example;

1	2	3	4	5	6	7	8	9	10
DRESP1	30		GSTRESS				ALL		

To tag the magnitude of user function of u velocities at filed point 12, 13, 15, and 17 the data entry is:

1	2	3	4	5	6	7	8	9	10
DRESP1	40		UFVELO			1	12	13	17
+	15								

Note that the PTYPE data is blank and that the filed point ID’s do not have to be in any particular order. The field point are defined in the UFDATA file.

3.1.5 Geometric Responses (DRESPG)

The geometric responses in the analysis model may be used to construct constraints or objective functions in the design model. The DRESPG input data is used to get geometric quantities for the design model. Currently, this data statement can be used to calculate the distance between points. The format of the DRESPG input data is:

1	2	3	4	5	6	7	8	9	10
DRESP G	ID	LABEL	RTYPE	Blank	Blank	Blank	Att1	Att2	Att3
+	Att4	Etc.							

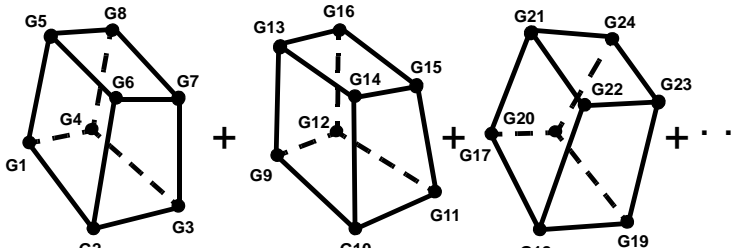
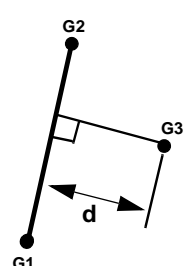
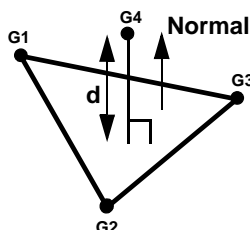
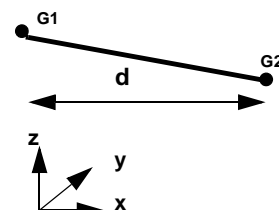
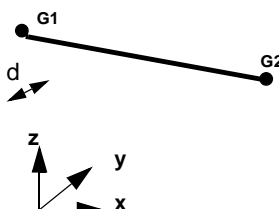
Each DRESPG data has a unique ID, which is required, and the optional LABEL can be used to identify it in the program output.

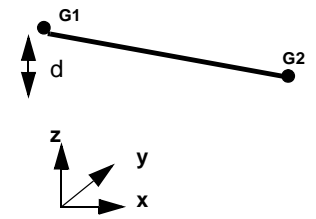
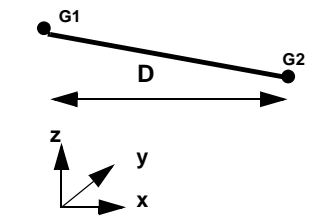
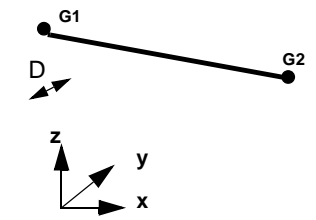
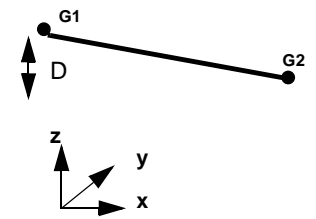
The response type (RTYPE) must be ANGLE, LENGTH, AREA3, AREA4, VOL4, VOL6, VOL8, DGLINE, DGPLANE, DIFFX, DIFFY, DIFFZ, DISTX, DISTY and DISTZ. The Atti are grid points.

The Atti are the grid IDs.

The following table illustrates the meaning of each response type.

RTYPE	DEFINITION (Gi used here is the grid ID, GIDi)
ANGLE	
LENGTH	
AREA3	
AREA4	
VOL4	
VOL6	

RTYPE	DEFINITION (Gi used here is the grid ID, GIDi)
VOL8	
DGLINE	 $d \geq 0$
DGPLANE	 <p>IF GID4 IS ON THE POSITIVE SIDE OF THE PLANE, $d > 0$</p> <p>IF GID4 IS ON THE NEGATIVE SIDE OF THE PLANE, $d < 0$</p>
DIFFX	 <p>IF GID2 IS ON THE POSITIVE X SIDE OF THE GID1, $d > 0$</p> <p>IF GID2 IS ON THE NEGATIVE X SIDE OF THE GID1, $d < 0$</p> <p>d is measured in the basic coordinate system</p>
DIFFY	 <p>IF GID2 IS ON THE POSITIVE Y SIDE OF THE GID1, $d > 0$</p> <p>IF GID2 IS ON THE NEGATIVE Y SIDE OF THE GID1, $d < 0$</p> <p>d is measured in the basic coordinate system</p>

DIFFZ	 <p>IF GID2 IS ON THE POSITIVE Z SIDE OF THE GID1, $d > 0$</p> <p>IF GID2 IS ON THE NEGATIVE Z SIDE OF GRID GID1, $d < 0$</p> <p>d is measured in the basic coordinate system</p>
DISTX	 <p>D is always positive or zero ($DISTX \geq 0$)</p> <p>D is measured in the basic coordinate system</p>
DISTY	 <p>D is always positive or zero ($DISTY \geq 0$)</p> <p>D is measured in the basic coordinate system</p>
DISTZ	 <p>D is always positive or zero ($DISTZ \geq 0$)</p> <p>D is measured in the basic coordinate system</p>

For example:

	1	2	3	4	5	6	7	8	9	10
DRESP G	101	LENGTH	LENGTH					10	11	12
+	13	14								

In this example the response 101 is calculated by adding the distance between grids 10 and 11, 11 and 12, 12 and 13, and 13 and 14.

The geometric responses tagged by DRESPG can be referenced by DOBJ, DINDEX, DMATCH, DMATCH2, DCONS, DRESP2 and DRESP3.

3.1.6 Objective Function (DOBJ)

When only one response is to be selected as the objective function. The DOBJ data entry can be used. When one or multiple responses are needed, the DINDEX data can be used.

The DOBJ input data is used to flag one of the responses (DRESP1, DRESP2, DRESP3, DRESPG or DRESPU) as the design objective function. The format of the DOBJ input data is:

1	2	3	4	5	6	7	8	9	10
DOBJ	RID	LABEL	LID	MIN/MAX					

The RID corresponds to the DRESP ID that is the objective function response. The optional LABEL is used to help identify the objective function in the program output. The LID corresponds to the LOADCASE or LOADCOM number in which the response will be the objective function (default is the first LOADCASE). Finally, the last data specifies whether the objective function is to be minimized or maximized (default is minimization). If the RID corresponds to a mass, volume, or DRESPU response then the LID is ignored.

For example, the DOBJ data for mass minimization could be:

1	2	3	4	5	6	7	8	9	10
DOBJ	10								

Here 10 is the DRESP1 ID that tags the system mass.

To maximize the response tagged on DRESP1 entry 33 in LOADCASE 5 the DOBJ data would be:

1	2	3	4	5	6	7	8	9	10
DOBJ	33	EL10STR	5	MAX					

3.1.7 Multi-objective Function (DINDEX)

The DINDEX data is used to specify the responses the user needs to be included in an index objective function. This is done by specifying DRESP IDs, weighting factors and LOADCASE numbers in the format:

1	2	3	4	5	6	7	8	9	10
DINDEX	RID1	LID1	W1	RID2	LID2	W2			
+	RID3	LID3	W3	-etc.-					

RID_i is the response number, LID_i is the load case number and W_i is the weighting factor.

DINDEX is an alternative way to select a design objective function. The index function is a weighted sum of response terms. How each term enters into the index function depends on the DOPT parameter, DINDEXM.

If DINDEXM is 0 or 1, then responses are normalized by their values in the first design cycle. If DINDEXM is 2 or 3, then responses are not normalized.

- DINDEXM = 0 or 1

$$R_i = \frac{\text{Response}_i}{|\text{Response}_{0i}|} \quad (\text{Eq. 3-8})$$

- DINDEXM = 2 or 3

$$R_i = \text{Response}_i \quad (\text{Eq. 3-9})$$

The index objective function is calculated using the following equation:

$$T = \sum_{i=1} f_i \quad (\text{Eq. 3-10})$$

If the DOPT parameter DINDEXM is 0 or 2, then the compliance index objective function terms are calculated using reciprocals for negative weighting factors. If DINDEXM is 1 or 3, then the compliance index objective function is a straight summation.

- DINDEXM = 0 or 2

$$f_i = \begin{cases} W_i \cdot R_i & \text{if } W_i > 0 \\ \frac{-W_i}{R_i} & \text{if } W_i < 0 \end{cases} \quad (\text{Eq. 3-11})$$

- DINDEXM = 1 or 3

$$f_i = W_i R_i \quad (\text{Eq. 3-12})$$

Typically, the index relationship is used for combining strain energies and/or frequency responses. In this case typically the weighting factor used are positive for the strain energy responses and negative for frequency responses.

The reason for this is that *GENESIS* will minimize the index function. The choice of a positive weighting factor for strain energy and negative for frequency will make the structure stiffer.

In any case, it is up to the user to decide the sign of the weighting factors.

DINDEX is not the only way to combine responses to create the design objective. A more general way can be obtained by using DOBJ referencing DRESP2, DRESP3 or DRESPU. The advantage of using DINDEX is that the data is simpler and that *GENESIS* provides a summary with the individual contributing responses. DINDEX is similar to the TINDEX data use in topology optimization.

DINDEX can be used together with DMATCH/DMATCH2 to simultaneously minimize and match responses.

Examples:

Minimize

$$TI = \frac{1.0 \times (\text{SEnergy}) (\text{load case 10})}{\text{SEnergy (load case 10 initial)}} + \frac{100 \times \text{Freq (mode 1, load case 20 initial)}}{\text{Freq (mode 1, load case 20)}}$$

The data for this problem could be:

	1	2	3	4	5	6	7	8	9	10
DRESP1	1	ENERGY	SENERGY							
DRESP1	2	MODE1	FREQ	1						
\$										
DINDEX	1	10	1.0							
+	2	20	-100.0							
DOPT										
+	DINDEX M	0								

Since the default value of the DINDEXM parameter is 0, the DOPT data statement is not required.

Instead of using the reciprocal of the frequencies in the compliance index function the user could use the negative components as it is shown in the next problem:

$$TI = \frac{1.0 \times (\text{SEnergy}) (\text{load case 10})}{\text{SEnergy (load case 10 initial)}} - \frac{100 \times \text{Freq (mode 1, load case 20)}}{\text{Freq (mode 1, load case 20 initial)}}$$

The data for this problem could be:

	1	2	3	4	5	6	7	8	9	10
DRESP1	1	ENERGY	SENERGY							
DRESP1	2	MODE1	FREQ	1						
\$										
DINDEX	1	10	1.0							
+	2	20	-100.0							
DOPT										
+	DINDEX M	1								

In this case the DOPT parameter DINDEXM is required to override the default value of 0.

$$\text{Minimize TI} = 1.0 * \text{Strain Energy (load case 10)} + \frac{100.0}{\text{Freq (mode 1, load case 20)}}$$

The data for this problem could be:

	1	2	3	4	5	6	7	8	9	10
DRESP1	1	ENERGY	SENERGY							
DRESP1	2	MODE1	FREQ	1						
\$										
DINDEX	1	10	1.0							
+	2	20	-100.0							
DOPT										
+	DINDEX M	2								

Instead of using the reciprocal of the frequencies in the compliance index function the user could use the negative components as it is shown in the next problem:

$$\text{TI} = 1.0 * \text{Strain Energy (load case 10)} - 100 * \text{Freq (mode 1, load case 20)}$$

The data for this problem could be:

	1	2	3	4	5	6	7	8	9	10
DRESP1	1	ENERGY	SENERGY							
DRESP1	2	MODE1	FREQ	1						
\$										
DINDEX	1	10	1.0							
+	2	20	-100.0							
DOPT										
+	DINDEX M	3								

In this case the DOPT parameter DINDEXM is required to override the default value of 0.

3.1.8 Constraints (DCONS and DCONS2)

The DCONS data is used to place lower and upper bound constraints on the DRESP tagged responses (DRESP1, DRESP2, DRESP3 or DRESPG). This is done by specifying the DRESP ID, LOADCASE or LOADCOM number, and lower and upper bound values in the format:

1	2	3	4	5	6	7	8	9	10
DCONS	RID	LID1	LB1	UB1	LID2	LB2	UB2		
+		LID3	LB3	UB3	...				

where LID_i is the load case number and LB_i and UB_i are the bounds for load case *i*.

Note that different bounds can be used in different load cases and that the response does not have to be constrained in all load cases. If the response has the same bounds in all the load cases then the following alternate form of the DCONS data can be used:

1	2	3	4	5	6	7	8	9	10
DCONS	RID	"ALL"	LB1	UB1					

If this form of the DCONS data is used in input data sets that contain static and/or frequency calculations and/or heat transfer and/or dynamic load cases, *GENESIS* will only constrain the response in the appropriate load cases (static responses in static load cases, frequency responses in frequency calculation load cases, dynamic responses in dynamic load cases and temperatures in heat transfer loadcases). If the DCONS data references a mass or volume response then the LID will be ignored.

The constraints can also be defined using the DCONS2 entry. The purpose of DCONS2 is to use constraint bounds relative to a response's initial value. The form of DCONS2 is similar to DCONS:

1	2	3	4	5	6	7	8	9	10
DCONS2	RID	LID1	LBF1	UBF1	LID2	LBF2	UBF2		
+		LID3	LBF3	UBF3	...				

where, LBF_i is a factor that scales the initial value of the response to calculate the actual lower bound and UBF_i is a factor that scales the initial value of the response to calculate the actual upper bound.

3.1.9 Matching Analysis Results (DMATCH, DMATCH2)

The DMATCH or DMATCH2 data is used to specify responses (DRESP1, DRESP2, DRESP3, DRESPG and DRESPU) that are to match target values. This is done by specifying DRESP ID, LOADCASE or LOADCOM number, target values and weighting factors for any number of responses in the format:

1	2	3	4	5	6	7	8	9	10
DMATCH2	RID1	LID1	T1	W1	RID2	LID2	T2	W2	
+	RID3	LID3	T3	W3	...				

where RID_i is the response number, LID_i is the load case number, T_i is the target value for the response, and W_i is the weighting factor.

DMATCH is similar to DMATCH2, with all W_i assumed to be 1.0:

1	2	3	4	5	6	7	8	9	10
DMATCH	RID1	LID1	T1	RID2	LID2	T2	RID3	LID3	T3
+	RID4	LID4	T4	...					

Note that each DRESP data referenced by DMATCH or DMATCH2 data can only generate one response per load case. To specify that the response described by DRESP 3 should be 0.3 in load case 1 and that the response described by DRESP5 should be 70.0 in load case 2 the data would be:

1	2	3	4	5	6	7	8	9	10
DMATCH	3	1	0.3	5	2	70.0			

If the DMATCH data is used then there can be no DOBJ or DMATCH2 data. DMATCH data can be used simultaneously with DINDEX to form a hybrid objective where some responses will be matched and other will be minimized. DCONS data can also be used in conjunction with DMATCH data.

To select the method to use for matching responses, use the parameter IMATCH. To select the Least Squares method use IMATCH = 0 and to select the Beta method use IMATCH = 1. The default is IMATCH = 0.

When some responses are more important to match than others, weighting factors can be specified using DMATCH2:

1	2	3	4	5	6	7	8	9	10
DMATCH2	3	1	0.3	1.0					
+	5	2	70.0	1.6					

3.1.10 Design of Laminated Composites

The layup of laminated composites can be optimized by *GENESIS*. Each composite can have up to 400 layers. The individual layer thicknesses and material orientation angles can be controlled by DVPROP1 and DVPROP2 data. The FID for a layer thickness is $100 + i$, where i is the layer number. For example, the FID for the thickness of layer 3 is 103 and the FID for layer 23 is 123. The FID for a layer material orientation angle is $500 + i$, where i is the layer number. For example, the FID for the material orientation angle of layer 4 is 504. The distance from the reference plane to the bottom of the first layer (Z_0) can also be designed. The FID for Z_0 is 4.

Optionally, multiple thicknesses and angles can be designed by a single **DVPROP4** entry. The DVPROP4 data provides a simpler way to design thicknesses and angles. It does not require the use of FID numbers, as the continuation line number identifies the layer number. When using DVPROP4 data, the user may also design Z_0 , GE and the non structural mass, if necessary, using DVPROP1 or DVPROP2.

The layup configurations is specified with the LAM data in the PCOMP/PCOMPG input data. There are three possible layup configurations: blank; where each layer is specified, SYM; where only the bottom layers of a symmetric layup are specified and the value of Z_0 is always $-1/2$ of the total thickness and, SYM1; where only the bottom layers of a symmetric layup are specified and the value of Z_0 must be specified by the user. If the SYM layup is used then no DVPROP1 or DVPROP2 data can reference Z_0 (FID=4). Note that if the blank or SYM1 layup is used when the layer thicknesses are being designed, the user will most likely want to use DVPROP1 or DVPROP2 data to control the value of Z_0 .

There are several responses for DRESP1 that can only be used with composite elements. These responses are identified by RTYPE values FINDEX, CSTRESS, CSTRAIN, CTHICK and LTHICK.

The composite ply failure index can be used as a response with DRESP1 data. The RTYPE for the composite ply failure index is FINDEX. The Findex Item Code (ATTA) can be 1 or 2. The item code for the composite ply failure is 1. This will create a composite ply failure index response for every layer (including symmetric layers) of the composite. The item code for interlaminar shear failure is 2. This will create a composite interlaminar shear response for every bonding interface (including symmetric layers) of the composite.

In the output files the composite ply failure index response item number is calculated as $10 * L + i$, where L is the layer number and i is the failure mode specified in the table below. For example, item number 34 is for layer 3 if when HOFF failure theory is used and the item number 57 is for layer 5 STRN theory failure in the 2 direction.

i	Ply Failure Mode
3	HILL
4	HOFF

5	TSAI
6	STRN in the 1 Direction
7	STRN in the 2 Direction
8	STRN in Shear

Stress and strain of individual layers can also be used as responses with DRESP1 data. The RTYPE for layer stress is CSTRESS, while the RTYPE for layer strain is CSTRAIN. In either case, the result is calculated at the center of the element, and halfway through the thickness of the layer.

The total thickness of a composite layup can be used as a response with DRESP1 data. The RTYPE for the total thickness response is CTHICK. The thickness of individual layers can also be used as a response with DRESP1. In this case, the RTYPE is LTHICK. The layer thickness item code (ATTA) can be 1 or 2. Item code 1 gives the actual layer thickness, while item code 2 gives the layer thickness fraction, which is defined as the layer thickness divided by the total thickness of the layup.

3.2 Basic Shape Design Capabilities

There are two approaches to shape design optimization available in *GENESIS*. The first is called the basis vector approach and is usually used with continuum structures (structures composed of plate/shell or solid elements). The second is called the grid perturbation approach and is most often used with truss and frame structures. However, either method may be used for any structure, depending on what is most convenient to the user. Both approaches use the **DVGRID** input data which defines the relationship between the design variables and the shape of the structure.

3.2.1 Basis Design Approach

This approach is used when the value of the design parameter **BASIS** is 1. In this approach the designer first generates several trial designs called basis vectors (vectors of grid point locations). *GENESIS* is then used to find the optimum linear combination of these basis vectors in the actual design. The design variables are used to control the participation of the basis design (basis vectors) in the actual design. The following relationship is used

$$\{XYZ\} = \{XYZ_0\} + DV_1\{XYZ_1 - XYZ_0\} + DV_2\{XYZ_2 - XYZ_0\} + DV_3\{XYZ_3 - XYZ_0\} + \dots \quad (\text{Eq. 3-13})$$

The $\{XYZ_0\}$ basis vector is composed of the grid point locations specified on the GRID data. The $\{XYZ_i\}$ basis vectors are constructed using the DVGRID input data which has the format:

1	2	3	4	5	6	7	8	9	10
DVGRID	DVID	GID	CID		N1	N2	N3	BASIS	

The location of grid point number GID is specified by N1, N2, and N3 in the CID coordinate system. The CID coordinate system is usually the same as the CP (input) coordinate system specified on the GRID data for this grid (blank or 0 references the basic system). The DVID is the design variable number that controls the participation of this grid point's location in the actual design.

If the location of a grid point is the same in a basis design and the original design, it does not have to be specified on a DVGRID data entry. *GENESIS* assumes that all missing grid points for each design variable are in the location specified on the GRID input data.

The engineer almost always wants to begin with the design specified on the GRID data. In this case the initial values of the shape design variables should be set equal to zero. Note that the shape design variables can have negative values.

If the engineer wishes to use nonzero starting design variable values but start the process with the shape specified on the GRID input data, the DLINK input data must be used. The zero valued dependent design variable, which is specified on the DVGRID data, is linked to be equal to the independent design variable minus its original value. Therefore,

$$DV_{\text{dep}} = DV_{\text{indep}} - DV_{\text{indep}_0} \quad (\text{Eq. 3-14})$$

For example:

1	2	3	4	5	6	7	8	9	10
---	---	---	---	---	---	---	---	---	----

Shape and Sizing Design Capabilities

DVAR	1	INDEP	3.0	-10.0	10.0				
DVAR	2	DEP	0.0	-13.0	7.0				
DLINK	2	-3.0		1	1.0				
DVGRID	2	20			1.0	2.0	0.0	1	
GRID	20		1.0	1.0	0.0				

In this example the independent design variable 1 controls the location of grid 20. The dependent design variable 2 has a value of zero in the initial design. Therefore the location of grid 20 is (1.0,1.0,0.0) in the initial design.

Because the general concept of using basis designs is somewhat complicated, the following more detailed discussion is offered.

Consider the equation used by *GENESIS* to update a design of n grids and m basis designs:

$$\begin{aligned}
 & \begin{Bmatrix} X_1 \\ Y_1 \\ Z_1 \\ X_2 \\ Y_2 \\ Z_2 \\ \vdots \\ \vdots \\ X_n \\ Y_n \\ Z_n \end{Bmatrix} = \begin{Bmatrix} X_1 \\ Y_1 \\ Z_1 \\ X_2 \\ Y_2 \\ Z_2 \\ \vdots \\ \vdots \\ X_n \\ Y_n \\ Z_n \end{Bmatrix}^0 + DV_1 \left\{ \begin{Bmatrix} X_1 \\ Y_1 \\ Z_1 \\ X_2 \\ Y_2 \\ Z_2 \\ \vdots \\ \vdots \\ X_n \\ Y_n \\ Z_n \end{Bmatrix}^1 - \begin{Bmatrix} X_1 \\ Y_1 \\ Z_1 \\ X_2 \\ Y_2 \\ Z_2 \\ \vdots \\ \vdots \\ X_n \\ Y_n \\ Z_n \end{Bmatrix}^0 \right\} \\
 & + DV_2 \left\{ \begin{Bmatrix} X_1 \\ Y_1 \\ Z_1 \\ X_2 \\ Y_2 \\ Z_2 \\ \vdots \\ \vdots \\ X_n \\ Y_n \\ Z_n \end{Bmatrix}^2 - \begin{Bmatrix} X_1 \\ Y_1 \\ Z_1 \\ X_2 \\ Y_2 \\ Z_2 \\ \vdots \\ \vdots \\ X_n \\ Y_n \\ Z_n \end{Bmatrix}^0 \right\} + \dots + DV_m \left\{ \begin{Bmatrix} X_1 \\ Y_1 \\ Z_1 \\ X_2 \\ Y_2 \\ Z_2 \\ \vdots \\ \vdots \\ X_n \\ Y_n \\ Z_n \end{Bmatrix}^m - \begin{Bmatrix} X_1 \\ Y_1 \\ Z_1 \\ X_2 \\ Y_2 \\ Z_2 \\ \vdots \\ \vdots \\ X_n \\ Y_n \\ Z_n \end{Bmatrix}^0 \right\}
 \end{aligned}
 \tag{Eq. 3-15}$$

where the subscript on the coordinates x, y and z represents the grid number.

A good way to understand this equation is to examine it from two different points of view. The first is to examine the effects of the equation grid by grid. The second is to examine the equation by studying separately the effect of each design variable on the total shape (this is done by selecting one design variable and setting the rest to zero).

Let's start the examination by studying a single grid. First, assume that we have one design variable and we want to study grid number 7. The equation for this case would be:

$$\begin{Bmatrix} x_7 \\ y_7 \\ z_7 \end{Bmatrix} = \begin{Bmatrix} x_7 \\ y_7 \\ z_7 \end{Bmatrix}^0 + DV_1 \left\{ \begin{Bmatrix} x_7 \\ y_7 \\ z_7 \end{Bmatrix}^1 - \begin{Bmatrix} x_7 \\ y_7 \\ z_7 \end{Bmatrix}^0 \right\} \quad (\text{Eq. 3-16})$$

where the vector $\begin{Bmatrix} x_7 \\ y_7 \\ z_7 \end{Bmatrix}^0$ is the vector that defines the location of grid 7 in the original

design, which is defined in the analysis input data (GRID data). The vector $\begin{Bmatrix} x_7 \\ y_7 \\ z_7 \end{Bmatrix}^1$

corresponds to the coordinates of grid 7 in basis design 1 (defined by DVGRID data),

and finally, the vector $\begin{Bmatrix} x_7 \\ y_7 \\ z_7 \end{Bmatrix}$ is the calculated vector that defines the location of grid

7 in the current design. The last equation represents a line that contains the vectors

$\begin{Bmatrix} x_7 \\ y_7 \\ z_7 \end{Bmatrix}^0$ and $\begin{Bmatrix} x_7 \\ y_7 \\ z_7 \end{Bmatrix}^1$, as shown in **Figure 3-2**.

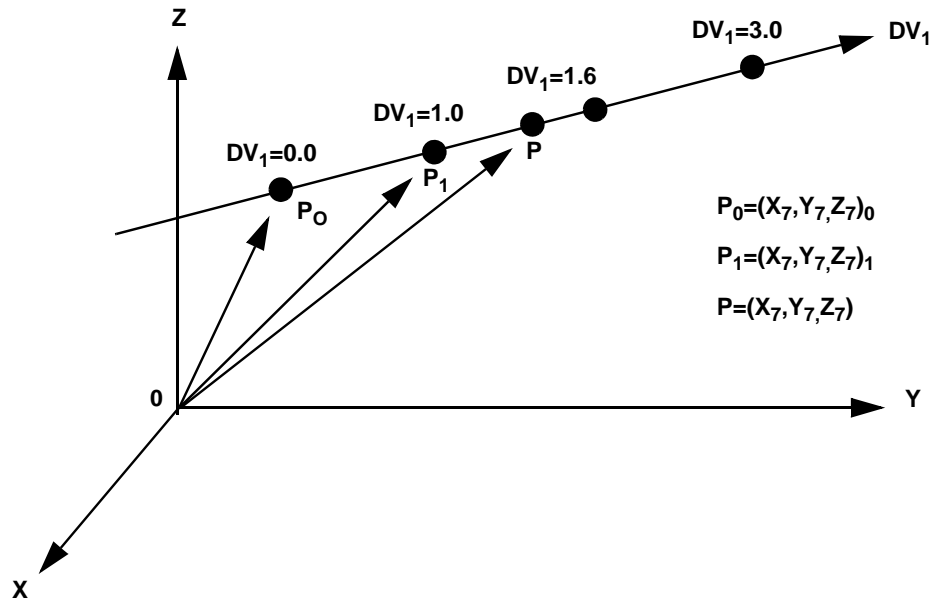


Figure 3-2 Single Grid

Any value of the design variable will produce a design in which the grid lies on the line. A value of 0.0 for the design variable will produce a design in which the position of grid 7 corresponds to its position in the original design. A value of 1.0 for the design variable corresponds to a design in which the position of grid 7 correspond to its position in basis vector 1.

A value of design variable 1 between 0.0 and 1.0 corresponds to a design in which the grid 7 is between its location in the original design and basis design 1. Finally, a value greater than 1.0 or smaller than 0.0 will produce a design in which grid 7 is not between the position of grid 7 in the original design and in basis design 1. In the figure above a final design with a value of 1.6 for the design variable 1 is shown.

Now, continue the study of a single grid but assume that we have two design variables with two basis designs. In this case the equation is:

$$\begin{Bmatrix} x_7 \\ y_7 \\ z_7 \end{Bmatrix} = \begin{Bmatrix} x_7 \\ y_7 \\ z_7 \end{Bmatrix}^0 + DV_1 \left\{ \begin{Bmatrix} x_7 \\ y_7 \\ z_7 \end{Bmatrix}^1 - \begin{Bmatrix} x_7 \\ y_7 \\ z_7 \end{Bmatrix}^0 \right\} + DV_2 \left\{ \begin{Bmatrix} x_7 \\ y_7 \\ z_7 \end{Bmatrix}^2 - \begin{Bmatrix} x_7 \\ y_7 \\ z_7 \end{Bmatrix}^0 \right\} \quad (\text{Eq. 3-17})$$

where the vector $\begin{Bmatrix} x_7 \\ y_7 \\ z_7 \end{Bmatrix}^0$ is the vector that defines the location of grid 7 in the original

design, the vectors $\begin{Bmatrix} x_7 \\ y_7 \\ z_7 \end{Bmatrix}^1$ and $\begin{Bmatrix} x_7 \\ y_7 \\ z_7 \end{Bmatrix}^2$ correspond to the coordinates of grid 7 in

basis designs 1 and 2 and finally, the vector $\begin{Bmatrix} x_7 \\ y_7 \\ z_7 \end{Bmatrix}$ is the vector that defines the

location of grid 7 in the current design. The last equation represent a plane that contains

the points $\begin{Bmatrix} x_7 \\ y_7 \\ z_7 \end{Bmatrix}^0$, $\begin{Bmatrix} x_7 \\ y_7 \\ z_7 \end{Bmatrix}^1$ and $\begin{Bmatrix} x_7 \\ y_7 \\ z_7 \end{Bmatrix}^2$, as in **Figure 3-3**.

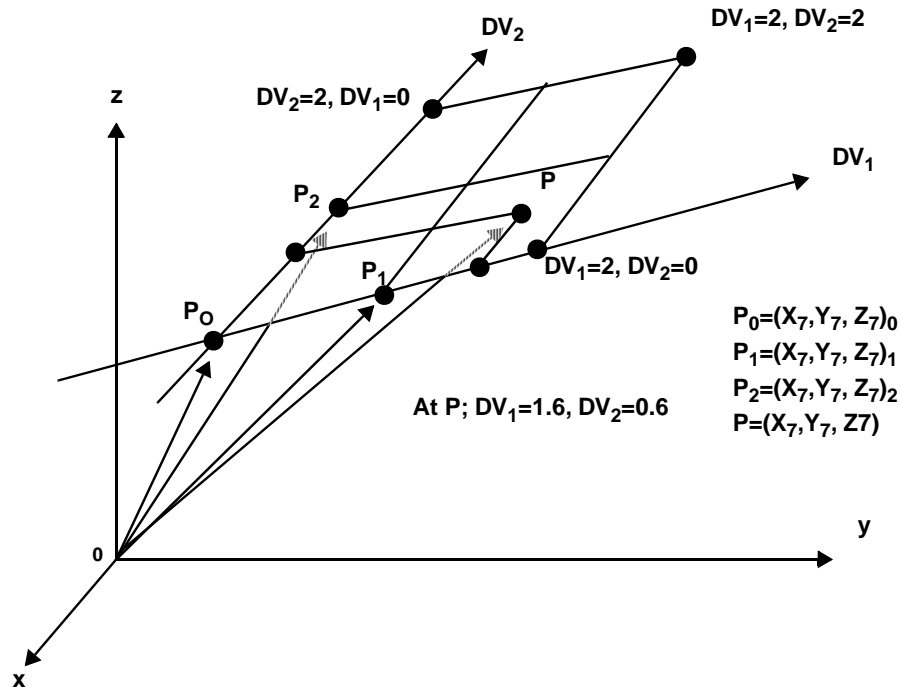


Figure 3-3 Two Design Variables

Any combination of the two design variables will produce a design for which this grid lies in the plane. A value of 0.0 for design variable 2 will produce a design in which the position of grid 7 is on the line defined by the position of grid 7 in the original design and the position of grid 7 in the basis design corresponding to design variable 1.

A value of 0.0 for design variable 1 and 1.0 for design variable 2 will produce a design with the shape of the basis design number 2. Similarly, a value of 0.0 for design variable 2 and 1.0 for design variable 1 will produce a design with the shape of basis design 1.

In the figure, the case where design variable 1 is 1.6 and design variable 2 is 0.6 is shown. This figure only shows the region of the plane defined by positive values of the design variables. However, the design variables can just as well be negative.

When more than two design variables are considered, the same approach is used (i.e., perturbation vectors are created that define the directions in which the grids can move). Then a linear combination of these perturbations is used to create a change in the grid position. Finally, the linear combination of perturbations is added to the initial position to create the current design.

$$\begin{Bmatrix} x_7 \\ y_7 \\ z_7 \end{Bmatrix} = \begin{Bmatrix} x_7 \\ y_7 \\ z_7 \end{Bmatrix}^0 + \sum_{i=1}^m DV_i \left\{ \begin{Bmatrix} x_7 \\ y_7 \\ z_7 \end{Bmatrix}^i - \begin{Bmatrix} x_7 \\ y_7 \\ z_7 \end{Bmatrix}^0 \right\} \quad (\text{Eq. 3-18})$$

Now, the manner in which all the grids move together will be studied. First, we will study the effect of just one design variable on the entire shape.

Assume that we want to study the effect of design variable number 1. In this case the equation for the entire shape would look like:

$$\begin{Bmatrix} X_1 \\ Y_1 \\ Z_1 \\ X_2 \\ Y_2 \\ Z_2 \\ \cdot \\ \cdot \\ X_n \\ Y_n \\ Z_n \end{Bmatrix} = \begin{Bmatrix} X_1 \\ Y_1 \\ Z_1 \\ X_2 \\ Y_2 \\ Z_2 \\ \cdot \\ \cdot \\ X_n \\ Y_n \\ Z_n \end{Bmatrix}^0 + DV_1 \left\{ \begin{Bmatrix} X_1 \\ Y_1 \\ Z_1 \\ X_2 \\ Y_2 \\ Z_2 \\ \cdot \\ \cdot \\ X_n \\ Y_n \\ Z_n \end{Bmatrix}^1 - \begin{Bmatrix} X_1 \\ Y_1 \\ Z_1 \\ X_2 \\ Y_2 \\ Z_2 \\ \cdot \\ \cdot \\ X_n \\ Y_n \\ Z_n \end{Bmatrix}^0 \right\} \quad (\text{Eq. 3-19})$$

Assume that $n=5$. The original design and basis design 1 are as shown in **Figure 3-4**.

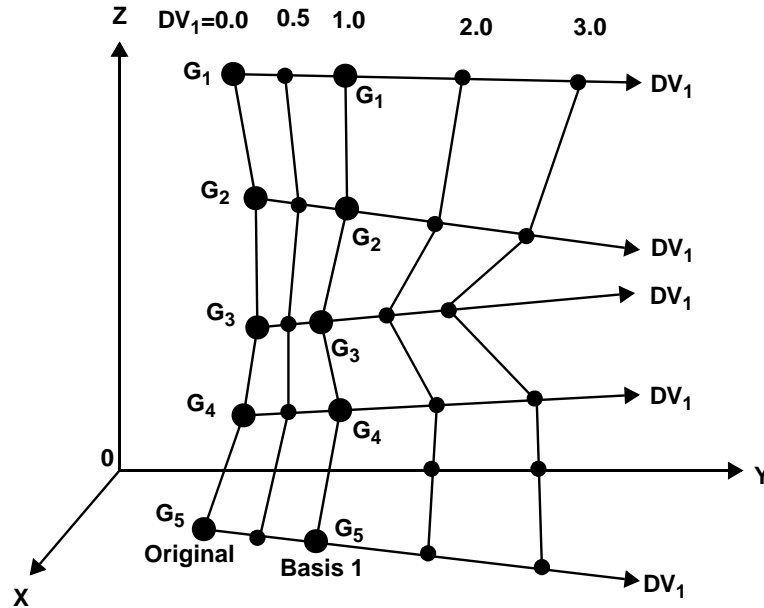


Figure 3-4Original Design and Basic Design

As we see from **Figure 3-4**, the shape can change from its original shape for a design variable value of 0.0 to another shape that is between the original and basis designs if the design variable value is between 0.0 and 1.0, or to a shape not between the original shape and basis design 1 for values of the design variable not in the 0.0 to 1.0 range. It should be pointed out that for a value of 1.0 of the design variable the shape corresponds to the shape of basis design 1. In the figure only the positive values of the design variable are shown. However, the design variable can have negative values.

Assume now that we add another basis vector to the problem studied before. The equation in this case would become;

(Eq. 3-20)

Figure 3-5.



Figure 3-5 Basic Design 2

In **Figure 3-5**, we see that the final shape is a linear combination of the perturbation vectors defined as the difference between the basis designs and the original design. For this case, we see that every grid is restricted to move in a plane.

3.2.2 Anomalies or Unexpected Results in Shape Optimization:

It is important to understand, geometrically, the effects that can be produced using basis designs. What follows it is a discussion of situations that the user must be aware of. The first one is related to changes of curvatures of shapes and the second relates to the reversal of grid locations.

Change of curvatures

From **Figure 3-6**, it can be seen that the original curvature does not change for $DV_1 < 1$, but for $DV_1 > 1$ the shape does change curvature. This is not necessarily bad, but the user should be aware that dramatic shape changes can be created with basis designs.

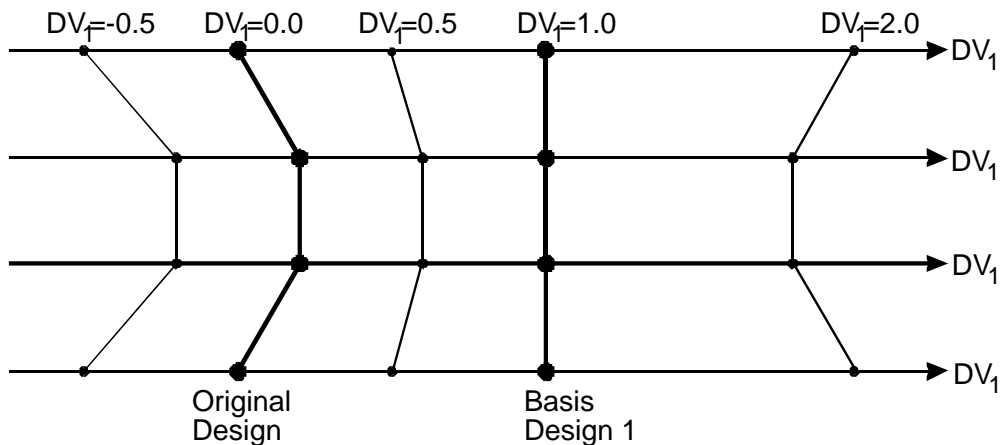


Figure 3-6 Change of Curvatures

Reversal of grid locations

The reversal of grid locations as shown in **Figure 3-7** can cause severe problems. This problem is produced because of a poor choice of positions of the grids in the basis design or because the design is allowed to change too much. In this case, if an element is connected to the two intermediate grids, severe ill-conditioning may result (or analysis may not be possible at all) for a value of DV_1 near, or greater than, 1.6.

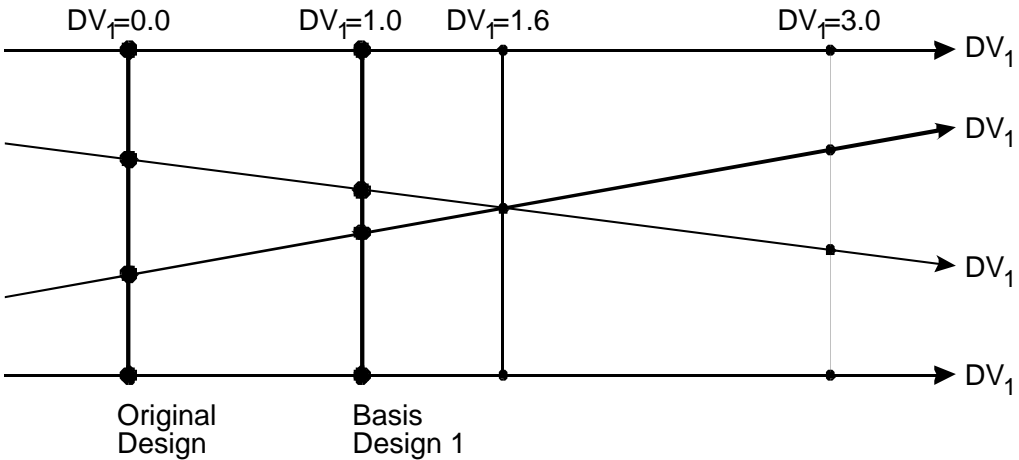


Figure 3-7 Reversal of Grid Locations

3.2.3 Grid Perturbation Approach

This approach is used when the value of the design parameter BASIS is 0. In this approach the design variable value determines the amount of perturbation in the location of the grid point. The direction of the grid perturbation is determined by the values of N1, N2, and N3 on the DVGRID data. When the grid perturbation approach is used the DVGRID input data has the format:

1	2	3	4	5	6	7	8	9	10
DVGRID	DVID	GID	CID	COEFF	N1	N2	N3	BASIS	

The grid perturbation vector is defined in the CID coordinate system (blank or 0 references the basic system). The perturbation vector is defined as $COEFF*(N1,N2,N3)$. The default value for COEFF is 1. The perturbation in the location of the grid point caused by design variable i is $DV_i*COEFF*(N1,N2,N3)$. During the optimization process the grid point locations are defined as:

$$\begin{aligned} \{XYZ\} = \{XYZ_0\} + DV_1\{XYZP_1\} + DV_2\{XYZP_2\} \\ + DV_3\{XYZP_3\} + \dots \end{aligned} \quad (\text{Eq. 3-21})$$

where the $\{XYZP_i\}$ are the vectors of grid perturbation for each design variable. If a grid is not referenced by a particular design variable with DVGRID data statement, then it is assumed that there is no grid perturbation for that design variable.

Usually this approach is used with one design variable for each degree of freedom involved in the shape design problem. For example to control the X and Y locations of GRIDs 5 and 6 the input data could be:

	1	2	3	4	5	6	7	8	9	10
DVGRID	1	5				1.0			0	
DVGRID	2	5					1.0		0	
DVGRID	3	6				1.0			0	
DVGRID	4	6					1.0		0	

Note that the values of the design variables can become negative during the optimization process. If the engineer desires that the initial grid point locations be the same as those specified on the GRID data, the shape design variables should be equal to zero.

The value of the coefficient (COEFF) should be chosen so that the design variable value is between -1.0 and 1.0. Of course it is impossible to know the final value of the design variable at the outset of the optimization process, but the engineer usually has a good idea about the possible range of the grid point locations. *GENESIS* will find the optimum design even if the final design variable value is 1.0E10, but the total number of design cycles may be larger than if the final value was 2.0.

Because the general concept of using perturbation designs is somewhat complicated, the following more detailed discussion is offered.

Consider the equation used to update a design of n grids and m perturbation vectors:

$$\begin{Bmatrix} X_1 \\ Y_1 \\ Z_1 \\ X_2 \\ Y_2 \\ Z_2 \\ \vdots \\ \vdots \\ X_n \\ Y_n \\ Z_n \end{Bmatrix} = \begin{Bmatrix} X_1 \\ Y_1 \\ Z_1 \\ X_2 \\ Y_2 \\ Z_2 \\ \vdots \\ \vdots \\ X_n \\ Y_n \\ Z_n \end{Bmatrix}^0 + DV_1 \begin{Bmatrix} p_1 \\ q_1 \\ r_1 \\ p_2 \\ q_2 \\ r_2 \\ \vdots \\ \vdots \\ p_n \\ q_n \\ r_n \end{Bmatrix}^1 + DV_2 \begin{Bmatrix} p_1 \\ q_1 \\ r_1 \\ p_2 \\ q_2 \\ r_2 \\ \vdots \\ \vdots \\ p_n \\ q_n \\ r_n \end{Bmatrix}^2 + \dots + DV_m \begin{Bmatrix} p_1 \\ q_1 \\ r_1 \\ p_2 \\ q_2 \\ r_2 \\ \vdots \\ \vdots \\ p_n \\ q_n \\ r_n \end{Bmatrix}^m \quad (\text{Eq. 3-22})$$

where $\begin{Bmatrix} X_i \\ Y_i \\ Z_i \end{Bmatrix}^0$, $i=1,2,\dots,n$ corresponds to the position of grid i in the initial design

and $\begin{Bmatrix} p_i \\ q_i \\ r_i \end{Bmatrix}^j$, $j=1,2,\dots,m$ corresponds to the perturbation of grid i associated with design variable j .

In this approach many of the entries for the perturbation vectors could be zero and the number of design variables could be large. The limiting case is that every grid has three design variables, each controlling only one perturbation vector in a different direction.

This equation can be understood by examining it grid by grid and then studying the effect of each design variable on the entire structure. Let's start by examining with a particular grid, say grid 7. First, assume that we have only one non zero perturbation vector associated with grid 7. The equation used to update this grid would look like:

$$\begin{Bmatrix} x_7 \\ y_7 \\ z_7 \end{Bmatrix} = \begin{Bmatrix} x_7 \\ y_7 \\ z_7 \end{Bmatrix}^0 + DV_1 \begin{Bmatrix} p_7 \\ q_7 \\ r_7 \end{Bmatrix}^1 \quad (\text{Eq. 3-23})$$

where $\begin{Bmatrix} x_7 \\ y_7 \\ z_7 \end{Bmatrix}^0$ is the position of the grid 7 in the initial design, which is defined in the

bulk data (GRID), $\begin{Bmatrix} p_7 \\ q_7 \\ r_7 \end{Bmatrix}^1$ is the perturbation vector, defined in the bulk data

(DVGRID), and $\begin{Bmatrix} x_7 \\ y_7 \\ z_7 \end{Bmatrix}$ is the position of grid 7 in the current design.

The last equation represents a line that contains the vector $\begin{Bmatrix} x_7 \\ y_7 \\ z_7 \end{Bmatrix}^0$ and is oriented in the direction $\begin{Bmatrix} p_7 \\ q_7 \\ r_7 \end{Bmatrix}^1$.

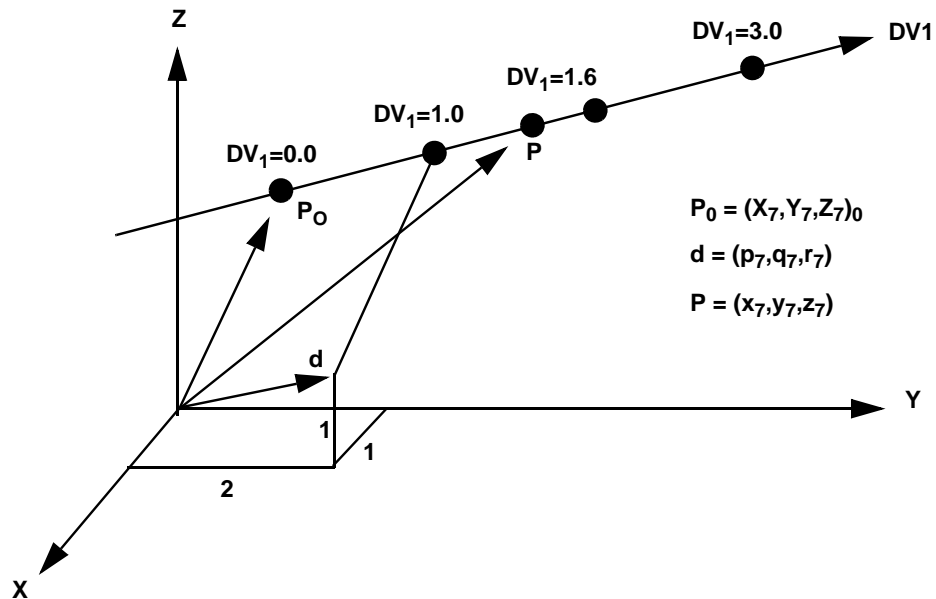


Figure 3-8 Perturbation Vector

Any value of the design variable will produce a design in which the grid is on the line. A value of 0.0 for DV1 represents a design in which grid 7 has not changed position. A value of 1.0 of DV1 is a design in which the grid has moved in the direction of the perturbation vector. The distance that the grid has moved for DV1 is exactly the magnitude of the perturbation vector. In general, for a positive value of DV1 the grid moves in the direction of the perturbation vector and the distance corresponds to the product of the design variable and the magnitude of the perturbation vector. For negatives values of the design variable the grid moves in the opposite direction.

Shape and Sizing Design Capabilities

The entry data for the perturbation vector in *GENESIS* is very flexible. The user can chose any coordinate system that has been defined. Also, *GENESIS* allows the use of an scaling factor to control the magnitude of the perturbation vector. The corresponding data is:

1	2	3	4	5	6	7	8	9	10
DVGRID	DVID	GID	CID	COEFF	N1	N2	N3	0	

In this case the data could be:

1	2	3	4	5	6	7	8	9	10
DVGRID	1	7		1.0	1.0	2.0	1.0	0	

In the data presented above we chose to use the default value (basic) for the coordinate

system that defines the $\begin{Bmatrix} N_1 \\ N_2 \\ N_3 \end{Bmatrix}$ vector. In this case the perturbation vector is calculated

in the basic coordinate system with the formula:

$$\begin{Bmatrix} p \\ q \\ r \end{Bmatrix} = [T] \begin{Bmatrix} N_1 \\ N_2 \\ N_3 \end{Bmatrix} = \begin{Bmatrix} 1.0 \\ 2.0 \\ 1.0 \end{Bmatrix} \quad (\text{Eq. 3-24})$$

where COEFF is the scaling factor, [T] is a coordinate transformation matrix between the coordinate system defined by CID and the Basic coordinate system and is generated

internally and $\begin{Bmatrix} N_1 \\ N_2 \\ N_3 \end{Bmatrix}$ is the vector used to define the direction of the perturbation

vector in the convenient CID coordinate system. In this example, as the $\begin{Bmatrix} N_1 \\ N_2 \\ N_3 \end{Bmatrix}$ vector

is in the basic coordinate system the [T] matrix corresponds to the identity matrix.

In this case a value of $DV_1=1.0$ represent a move of $\sqrt{1.0^2 + 2.0^2 + 1.0^2} = 2.45$ of grid 7 in the direction of the perturbation vector. If we wish to have a move of 1.0 corresponding a value of 1.0 for design variable 1, we can set the scaling coefficient COEFF in the DVGRID to $\frac{1.0}{2.45} \approx 0.40285$. In this case the data would look like:

	1	2	3	4	5	6	7	8	9	10
DVGRID	1	7			0.40825	1.0	2.0	1.0	0	

The above data is completely equivalent to the following:

	1	2	3	4	5	6	7	8	9	10
DVGRID	1	7			1.0	0.40825	0.81650	0.40825	0	

This represents the following line:

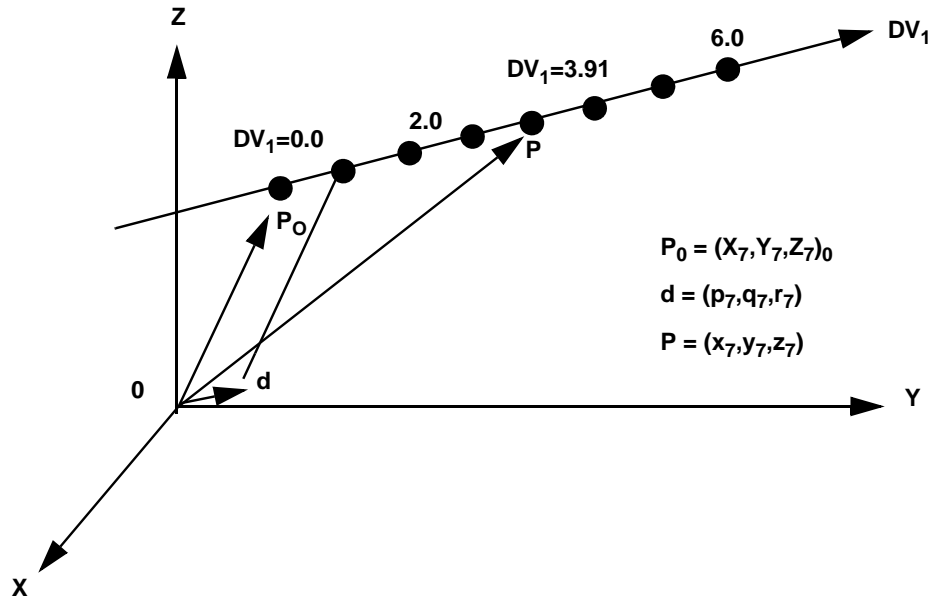


Figure 3-9 Perturbation Vector 1

Of course it is the same line because only the perturbation vector has been scaled. But, in this case the same design as with the first data would be a design with $DV_1=3.91$.

The position of grid 7 is calculated in the first case as:

$$\begin{Bmatrix} x_7 \\ y_7 \\ z_7 \end{Bmatrix} = \begin{Bmatrix} x_7 \\ y_7 \\ z_7 \end{Bmatrix}^0 + 1.6 \begin{Bmatrix} 1.0 \\ 2.0 \\ 1.0 \end{Bmatrix} = \begin{Bmatrix} x_7 \\ y_7 \\ z_7 \end{Bmatrix}^0 + \begin{Bmatrix} 1.6 \\ 3.2 \\ 1.6 \end{Bmatrix} \quad (\text{Eq. 3-25})$$

while in the second case the position grid 7 is calculated as:

$$\begin{Bmatrix} x_7 \\ y_7 \\ z_7 \end{Bmatrix} = \begin{Bmatrix} x_7 \\ y_7 \\ z_7 \end{Bmatrix}^0 + 3.91 \begin{Bmatrix} 0.40825 \\ 0.81650 \\ 0.40825 \end{Bmatrix} = \begin{Bmatrix} x_7 \\ y_7 \\ z_7 \end{Bmatrix}^0 + \begin{Bmatrix} 1.6 \\ 3.2 \\ 1.6 \end{Bmatrix} \quad (\text{Eq. 3-26})$$

From these two possible forms of the data we see that to geometrically interpret each perturbation it is necessary to represent both its direction and magnitude. The direction of the perturbation can be represented with a line that contains the initial design and is oriented in the direction of the perturbation. The magnitude of the perturbation can be represented on the line with a value of 1.0 for the design variable.

Now, continue the study of a single grid but assume that there are two perturbation vectors associated with the grid. Again assume that we are studying grid 7. In this case the equation could look like:

$$\begin{Bmatrix} x_7 \\ y_7 \\ z_7 \end{Bmatrix} = \begin{Bmatrix} x_7 \\ y_7 \\ z_7 \end{Bmatrix}^0 + DV_1 \begin{Bmatrix} p_7 \\ q_7 \\ r_7 \end{Bmatrix}^1 + DV_2 \begin{Bmatrix} p_7 \\ q_7 \\ r_7 \end{Bmatrix}^2 \quad (\text{Eq. 3-27})$$

where $\begin{Bmatrix} x_7 \\ y_7 \\ z_7 \end{Bmatrix}^0$ is the vector that defines the location of grid 7 in the initial design and

the vectors $\begin{Bmatrix} p_7 \\ q_7 \\ r_7 \end{Bmatrix}^1$ and $\begin{Bmatrix} p_7 \\ q_7 \\ r_7 \end{Bmatrix}^2$ correspond to the perturbation vectors associated to

grid 7 and design variables DV1 and DV2. The vector $\begin{Bmatrix} x_7 \\ y_7 \\ z_7 \end{Bmatrix}$ defines the location of the

grid 7 in the current design.

The last equation represents a plane that contains the point $\begin{Bmatrix} x_7 \\ y_7 \\ z_7 \end{Bmatrix}^0$. The defining

vectors of the plane are the perturbation vectors.

Any value of the two design variables will produce a design that is contained in the plane. A value of 0.0 for design variable 2 will produce a design in which the position of grid 7 is on the line defined by the position of grid 7 of the original design and perturbation vector 1.

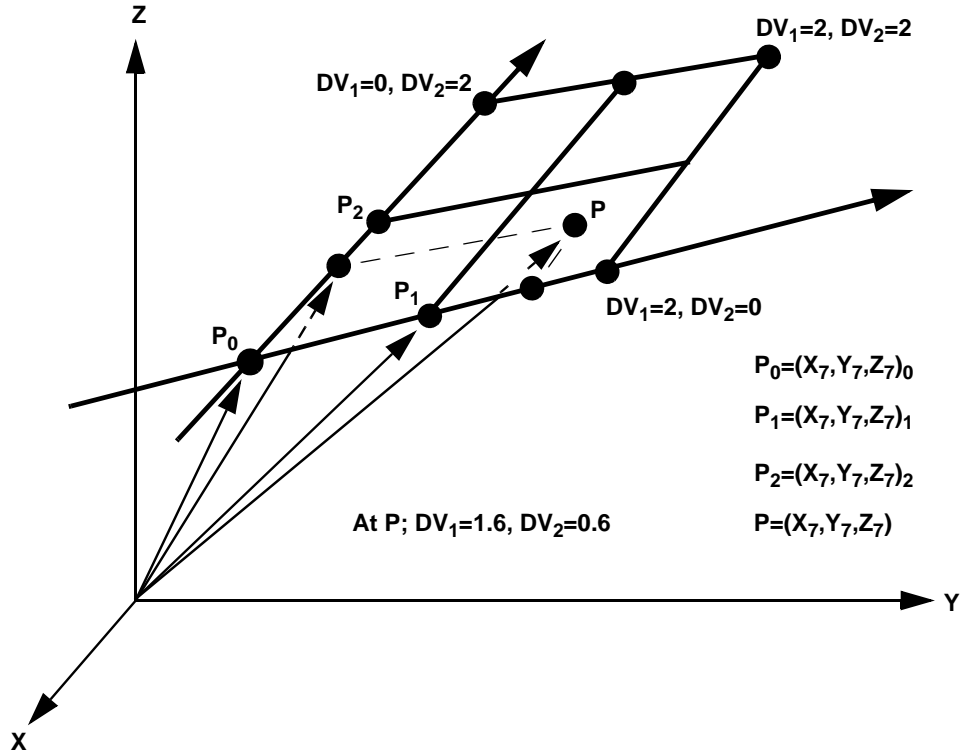


Figure 3-10 Perturbation Vector 2

In **Figure 3-10**, the case where design variable 1 is 1.6 and design variable 2 is 0.6 is shown. In this figure only the region of the plane defined by positive values of the design variables is shown. However, the design variables can be negative.

The data for the two variable example above could be

	1	2	3	4	5	6	7	8	9	10
DVGRID	1	7			1.0	1.0	2.0	1.0	0	
DVGRID	2	7			1.0	-1.0	1.7	1.6	0	

When more than two design variables are used the same approach is used by *GENESIS*, i.e. a linear combination of the perturbations is added to the initial position to create the current design.

$$\begin{Bmatrix} x_7 \\ y_7 \\ z_7 \end{Bmatrix} = \begin{Bmatrix} x_7 \\ y_7 \\ z_7 \end{Bmatrix}^0 + \sum_{i=1}^m DV_i \begin{Bmatrix} p_7 \\ q_7 \\ r_7 \end{Bmatrix}^i \quad (\text{Eq. 3-28})$$

Now the manner in which all grids move together will be studied. First, we will consider the effect of just one design variable on the entire shape.

Assume that we want to study the effect of design variable number 1. In this case the equation for the entire shape could look like:

$$\begin{Bmatrix} X_1 \\ Y_1 \\ Z_1 \\ X_2 \\ Y_2 \\ Z_2 \\ \cdot \\ \cdot \\ X_n \\ Y_n \\ Z_n \end{Bmatrix} = \begin{Bmatrix} X_1 \\ Y_1 \\ Z_1 \\ X_2 \\ Y_2 \\ Z_2 \\ \cdot \\ \cdot \\ X_n \\ Y_n \\ Z_n \end{Bmatrix}^0 + DV_1 \begin{Bmatrix} 0.0 \\ 0.0 \\ 0.0 \\ p_2 \\ q_2 \\ r_2 \\ \cdot \\ \cdot \\ 0.0 \\ 0.0 \\ 0.0 \end{Bmatrix}^1$$

Assume that $n=5$ and the original design and the perturbation is as shown **Figure 3-11**.

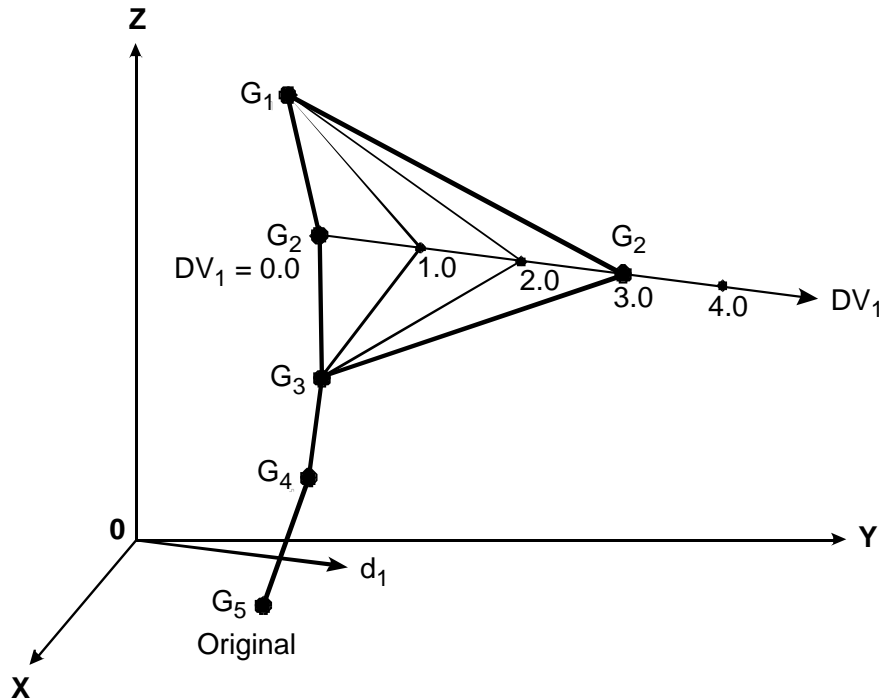


Figure 3-11 Perturbation Vector 1

As we see from **Figure 3-11**, the shape can change from its original shape, with design variable equal to 0.0, to another shape by moving the grid in the direction of the perturbation vector. As was mentioned before, in the perturbation approach, usually most of the entries of the perturbation vectors corresponding to one design variable are 0.0. This is not a limitation in *GENESIS*, in fact we can create a problem in which one design variable can control each grid of the structure, just as in the Basis vector approach. In this case the equation could look like **(Eq. 3-29)**.

$$\begin{Bmatrix} X_1 \\ Y_1 \\ Z_1 \\ X_2 \\ Y_2 \\ Z_2 \\ \vdots \\ \vdots \\ X_n \\ Y_n \\ Z_n \end{Bmatrix} = \begin{Bmatrix} X_1 \\ Y_1 \\ Z_1 \\ X_2 \\ Y_2 \\ Z_2 \\ \vdots \\ \vdots \\ X_n \\ Y_n \\ Z_n \end{Bmatrix}^0 + DV_1 \begin{Bmatrix} p_1 \\ q_1 \\ r_1 \\ p_2 \\ q_2 \\ r_2 \\ \vdots \\ \vdots \\ p_n \\ q_n \\ r_n \end{Bmatrix}^1 \quad (\text{Eq. 3-29})$$

The entire shape of the structure for different values of the design variable could look like:

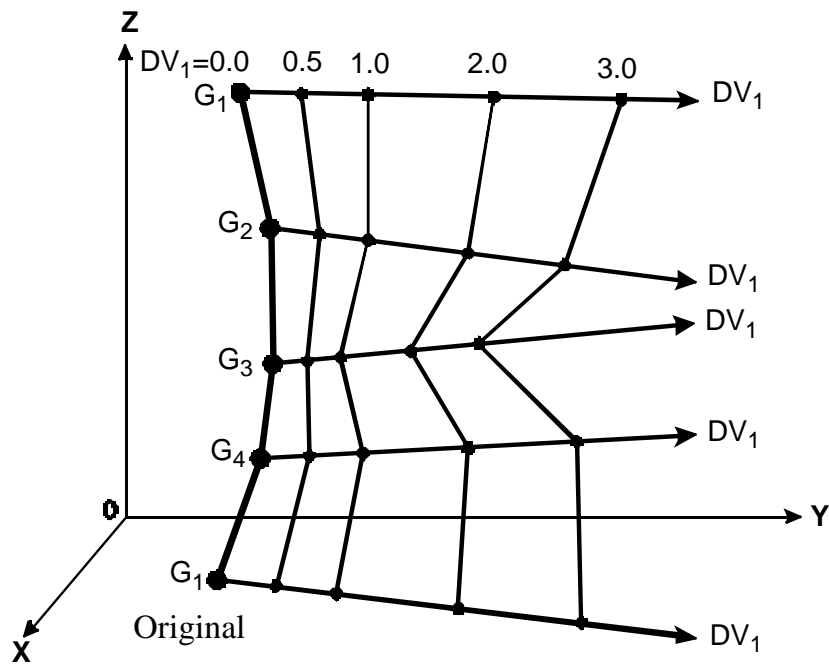


Figure 3-12 Perturbation Vector 2

In [Figure 3-12](#), the perturbation vectors associated with the grids are implicitly defined by the original design and a design variable value of 1.0. In the figure, four possible shapes besides the original shape are presented.

$$\begin{Bmatrix} X_1 \\ Y_1 \\ Z_1 \\ X_2 \\ Y_2 \\ Z_2 \\ \vdots \\ \vdots \\ X_n \\ Y_n \\ Z_n \end{Bmatrix} = \begin{Bmatrix} X_1 \\ Y_1 \\ Z_1 \\ X_2 \\ Y_2 \\ Z_2 \\ \vdots \\ \vdots \\ X_n \\ Y_n \\ Z_n \end{Bmatrix}^0 + DV_1 \begin{Bmatrix} 0.0 \\ 0.0 \\ 0.0 \\ p_2 \\ q_2 \\ r_2 \\ \vdots \\ \vdots \\ 0.0 \\ 0.0 \\ 0.0 \end{Bmatrix}^1 + DV_2 \begin{Bmatrix} 0.0 \\ 0.0 \\ 0.0 \\ p_2 \\ q_2 \\ r_2 \\ \vdots \\ \vdots \\ 0.0 \\ 0.0 \\ 0.0 \end{Bmatrix}^2 \quad (\text{Eq. 3-30})$$
[illegible]

GENESIS

In **Figure 3-13**, each of the two design variables controlled just one grid. Another case could be that each design variable controls all the grids just as we studied before for one design variable. In this case the equation used by *GENESIS* to update the grids would look like:

$$\begin{Bmatrix} X_1 \\ Y_1 \\ Z_1 \\ X_2 \\ Y_2 \\ Z_2 \\ \cdot \\ \cdot \\ X_n \\ Y_n \\ Z_n \end{Bmatrix} = \begin{Bmatrix} X_1 \\ Y_1 \\ Z_1 \\ X_2 \\ Y_2 \\ Z_2 \\ \cdot \\ \cdot \\ X_n \\ Y_n \\ Z_n \end{Bmatrix}^0 + DV_1 \begin{Bmatrix} p_1 \\ q_1 \\ r_1 \\ p_2 \\ q_2 \\ r_2 \\ \cdot \\ \cdot \\ p_n \\ q_n \\ r_n \end{Bmatrix}^1 + DV_2 \begin{Bmatrix} p_1 \\ q_1 \\ r_1 \\ p_2 \\ q_2 \\ r_2 \\ \cdot \\ \cdot \\ p_n \\ q_n \\ r_n \end{Bmatrix}^2 \quad (\text{Eq. 3-31})$$

and the structure could look like:

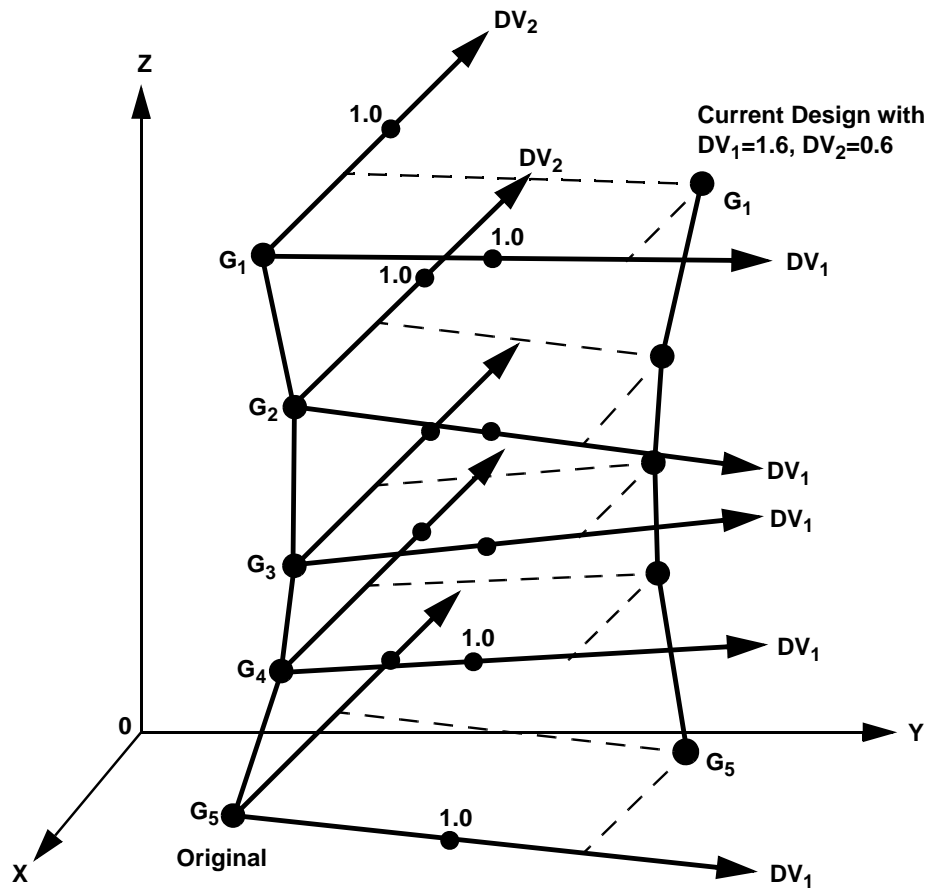


Figure 3-14 Perturbation Vectors 1 and 2

In **Figure 3-14**, we see that the current shape is a linear combination of the perturbation vectors and the original shape.

Comparison of Basis Design and Perturbation Approaches.

Using the Basis design approach, the fundamental equation used here is

$$\begin{aligned}
 & \begin{Bmatrix} X_1 \\ Y_1 \\ Z_1 \\ X_2 \\ Y_2 \\ Z_2 \\ \cdot \\ \cdot \\ X_n \\ Y_n \\ Z_n \end{Bmatrix} = \begin{Bmatrix} X_1 \\ Y_1 \\ Z_1 \\ X_2 \\ Y_2 \\ Z_2 \\ \cdot \\ \cdot \\ X_n \\ Y_n \\ Z_n \end{Bmatrix}^0 + DV_1 \left\{ \begin{Bmatrix} X_1 \\ Y_1 \\ Z_1 \\ X_2 \\ Y_2 \\ Z_2 \\ \cdot \\ \cdot \\ X_n \\ Y_n \\ Z_n \end{Bmatrix}^1 - \begin{Bmatrix} X_1 \\ Y_1 \\ Z_1 \\ X_2 \\ Y_2 \\ Z_2 \\ \cdot \\ \cdot \\ X_n \\ Y_n \\ Z_n \end{Bmatrix}^0 \right\} \\
 & + DV_2 \left\{ \begin{Bmatrix} X_1 \\ Y_1 \\ Z_1 \\ X_2 \\ Y_2 \\ Z_2 \\ \cdot \\ \cdot \\ X_n \\ Y_n \\ Z_n \end{Bmatrix}^2 - \begin{Bmatrix} X_1 \\ Y_1 \\ Z_1 \\ X_2 \\ Y_2 \\ Z_2 \\ \cdot \\ \cdot \\ X_n \\ Y_n \\ Z_n \end{Bmatrix}^0 \right\} + \dots + DV_m \left\{ \begin{Bmatrix} X_1 \\ Y_1 \\ Z_1 \\ X_2 \\ Y_2 \\ Z_2 \\ \cdot \\ \cdot \\ X_n \\ Y_n \\ Z_n \end{Bmatrix}^m - \begin{Bmatrix} X_1 \\ Y_1 \\ Z_1 \\ X_2 \\ Y_2 \\ Z_2 \\ \cdot \\ \cdot \\ X_n \\ Y_n \\ Z_n \end{Bmatrix}^0 \right\}
 \end{aligned}
 \tag{Eq. 3-32}$$

Similarly, using the grid perturbation approach,

$$\begin{Bmatrix} X_1 \\ Y_1 \\ Z_1 \\ X_2 \\ Y_2 \\ Z_2 \\ \cdot \\ \cdot \\ X_n \\ Y_n \\ Z_n \end{Bmatrix} = \begin{Bmatrix} X_1 \\ Y_1 \\ Z_1 \\ X_2 \\ Y_2 \\ Z_2 \\ \cdot \\ \cdot \\ X_n \\ Y_n \\ Z_n \end{Bmatrix}^0 + DV_1 \begin{Bmatrix} p_1 \\ q_1 \\ r_1 \\ p_2 \\ q_2 \\ r_2 \\ \cdot \\ \cdot \\ p_n \\ q_n \\ r_n \end{Bmatrix}^1 + DV_2 \begin{Bmatrix} p_1 \\ q_1 \\ r_1 \\ p_2 \\ q_2 \\ r_2 \\ \cdot \\ \cdot \\ p_n \\ q_n \\ r_n \end{Bmatrix}^2 + \dots + DV_m \begin{Bmatrix} p_1 \\ q_1 \\ r_1 \\ p_2 \\ q_2 \\ r_2 \\ \cdot \\ \cdot \\ p_n \\ q_n \\ r_n \end{Bmatrix}^m \quad (\text{Eq. 3-33})$$

3

In general, these two equations are equivalent, with

$$\begin{Bmatrix} p_1 \\ q_1 \\ r_1 \\ p_2 \\ q_2 \\ r_2 \\ \cdot \\ \cdot \\ p_n \\ q_n \\ r_n \end{Bmatrix}^i = \begin{Bmatrix} X_1 \\ Y_1 \\ Z_1 \\ X_2 \\ Y_2 \\ Z_2 \\ \cdot \\ \cdot \\ X_n \\ Y_n \\ Z_n \end{Bmatrix}^i - \begin{Bmatrix} X_1 \\ Y_1 \\ Z_1 \\ X_2 \\ Y_2 \\ Z_2 \\ \cdot \\ \cdot \\ X_n \\ Y_n \\ Z_n \end{Bmatrix}^0 \quad (\text{Eq. 3-34})$$

Thus, a perturbation vector may be thought of as simply the difference between two basis vectors. The two methods can be mixed in a given design task. The choice of methods is provided as a user convenience. In general, the basis design method is best if we have several candidate designs which are known to be reasonable, while the perturbation method is best if we only want to move one or a few grid locations.

3.3 Advanced Design Capabilities

The advanced design capabilities available in *GENESIS* include general nonlinear design variable to analysis model property relationships, generation of general nonlinear responses for objective or constraint functions, automatic generation of stress constraints, and user control over the constraint screening and optimization parameters.

3.3.1 General Nonlinear Design Variable to Property Relationships (DVPROP2)

In many design problems the analysis model properties are nonlinear functions of the design variables. This is especially true in frame design, where the beam section properties are almost always nonlinear functions of the cross sectional dimensions, and plate bending problems, where the bending stiffness is a nonlinear function of the plate thickness. This nonlinear relationship can be written in the form of a Fortran-like function and used to link the properties to the design variables. This is accomplished using the **DVPROP2**, **DEQATN**, and **DTABLE** input data. The format of the DVPROP2 data is:

	1	2	3	4	5	6	7	8	9	10
DVPROP2	ID	PID	FID	EQID						
+	"DVAR"	DVID1	DVID2	DVID3	...					
+	Blank	DVIDi								
+	"DTABLE"	CID1	CID2	CID3	...					
+	Blank	CIDi								

The ID, PID, and FID are the same as for the DVPROP1 data. The ID should be unique for all DVPROP2 data. The PID is the property ID and the FID is a code that specifies which element property is being updated. The EQID is the ID of the DEQATN data that contains the nonlinear relationship function. The DVIDi and CIDi are ID's of the design variables and constants that are arguments for this function. For example consider the relationship between the area moment of inertia of a rectangular beam and its cross sectional dimensions B and H. This relationship is

$$I = I(B, H) = \frac{BH^3}{12} \quad (\text{Eq. 3-35})$$

This relationship function is input using the DEQATN data as

	1	2	3	4	5	6	7	8	9	10
DEQATN	10	I(B,H) = B*H**3/12.0								

Shape and Sizing Design Capabilities

where the format for the DEQATN data is

1	2	3	4	5	6	7	8	9	10
DEQATN	EQID	F(ARG1, ARG2, ...) = ...							
+	...								

Note that the DEQATN data has only three pieces of information: the DEQATN keyword; the equation ID; and the equation itself. The equation can be arbitrarily long, and uses a Fortran-like syntax. An in-depth discussion of the equation utility is given in Section 5.1. The DVAR and DVPROP2 data for this example would be

1	2	3	4	5	6	7	8	9	10
DVAR	1	HEIGHT	5.0	0.1	20.0				
DVAR	2	BASE	2.0	0.1	6.0				
DVPROP2	100	20	4	10					
+	DVAR	2	1						

Note that DVPROP2 100 updates I1 on PBAR 20 using DEQATN 10 and design variables 1 and 2. The order that the design variables appear on the DVPROP2 data must be the same as the order that they appear in the parentheses on the DEQATN data. The DVPROP2 data generates the functional relationship

$$I1_{\text{PBAR20}} = \frac{DV_2 DV_1^3}{12.0} \quad (\text{Eq. 3-36})$$

In many problems it is convenient to use tabled constants in the functions. In the above example the number 12.0 could be replaced with a constant. This is shown in the following data.

1	2	3	4	5	6	7	8	9	10
DTABLE	TWELVE	12.0							
DVPROP2	100	20	4	10					
+	DVAR	2	1						
DEQATN	10	I(B,H) = B*H**3/TWELVE							

In some cases the use of the DVPROP2 data may require the generation of very complex equations. For example, the calculation of A, I1, I2, and J for a thin walled box section with four design variables. In order to lessen and simplify the input data for beam and plate bending elements a library of common cross sections has been built into *GENESIS*. This library is accessed using the DVPROP3 data.

3.3.2 Element Library Design Variable to Property Relationships (DVPROP3)

The design variable (cross sectional dimension) to analysis model property (section property) relationships are quite complex for many beam element cross sections. The generation of a DVPROP2 data entries and their corresponding DEQATN data for all the section properties and stress recovery points can be quite a large task. The **DVPROP3** data entry can be used to easily formulate these relationships. The DVPROP3 data references a built in library of common beam and plate cross sections. This design element library includes solid square, circular, and rectangular beams, box beams, I beams, solid plates, and sandwich plates to name a few. The complete library of cross sections, the formulae for their section properties, and stress recovery formulation are given at the end of this section. The DVPROP3 data is used to specify the cross section type and the design variables that control its shape. The format for the DVPROP3 data is:

1	2	3	4	5	6	7	8	9	10
DVPROP3	ID	PID	ELTYPE	ISHEAR					
+	CSD1	CSD2	CSD3	CSD4	...				

The ID is the unique DVPROP3 identifier. The PID is the property number of the PBAR or PSHELL data that is being updated. The ELTYPE is the type of cross section (square, box, I, etc.). The switch ISHEAR is used to choose whether or not to include shear deformation in the stiffness formulation of the BAR and plate/shell elements. The default is to not include shear deformation. The CSDi are used to determine the dimensions of the cross section. Either a design variable ID or a constant value is given for each cross sectional dimension. The number of CSDi must be equal to the number of cross sectional dimensions for the particular cross section. For example consider a solid rectangular section whose height is controlled by design variable 1 and base width by design variable 2. The DVPROP3 data for this example would be (see the tables at the end of this section for design variable definition):

1	2	3	4	5	6	7	8	9	10
DVPROP3	10	20	RECT						
+	2	1							

Shape and Sizing Design Capabilities

Note that the DVPROP3 ID is 10, the PBAR ID is 20, the element type is RECTangular, and no shear deformation is included (the default). All of the section properties for PBAR 10 are updated using this simple data entry. Note also that if it is desired to keep the base at a constant width of 5.0 the data would have the form:

	1	2	3	4	5	6	7	8	9	10
DVPROP3		10	20	RECT						
+		5.0	1							

All DVPROP3 data must reference at least one design variable ID.

Library of Bar and Plate Cross Sections (DVPROP3)

The set of cross sections available in the *GENESIS* bar cross section library are described here. The purpose of the bar cross section library is to simplify the design data by internally calculating information that would otherwise have to be provided by the user.

Stress constraints can be automatically generated for each element. See **DVPROP3** (p. 621). Note, that you also have the option of creating your own elements and linking them with *GENESIS*.

The bar element orientation and coordinate system is shown below. In the figures defining the elements, the stress calculation points are shown a positive face. That is, the stresses defined here are on the positive X face of the element at end b. At end a, the same definitions apply by considering the positive face at $X=0+$ (to visualize these stress locations at end a, look at the positive face of the element at a very small value of X).

The stresses are defined as positive if their directions are coincident with the forces that produce them. For example, longitudinal stress is positive if F_x is positive (traction stress is positive).

NOTE: In the following cross-section figures, the design variables are chosen to insure that the optimization will produce a buildable section. This requires that we treat “internal” dimensions as design variables. If the total height H (for example) of a box beam must not exceed a specified value, H_{\max} , this can be done by using a DRESP2 data statement to limit the overall dimension $h+2*t$ to the specified value, H_{\max} (see the corresponding figure for BOX3 below).

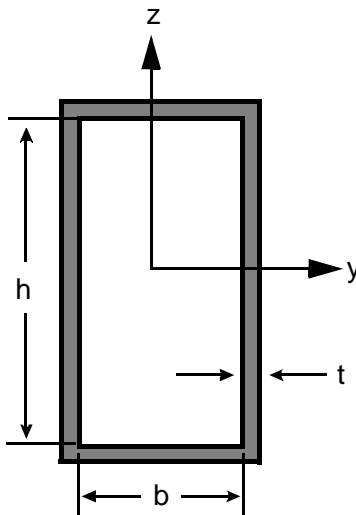


Figure 3-15 Box Beam

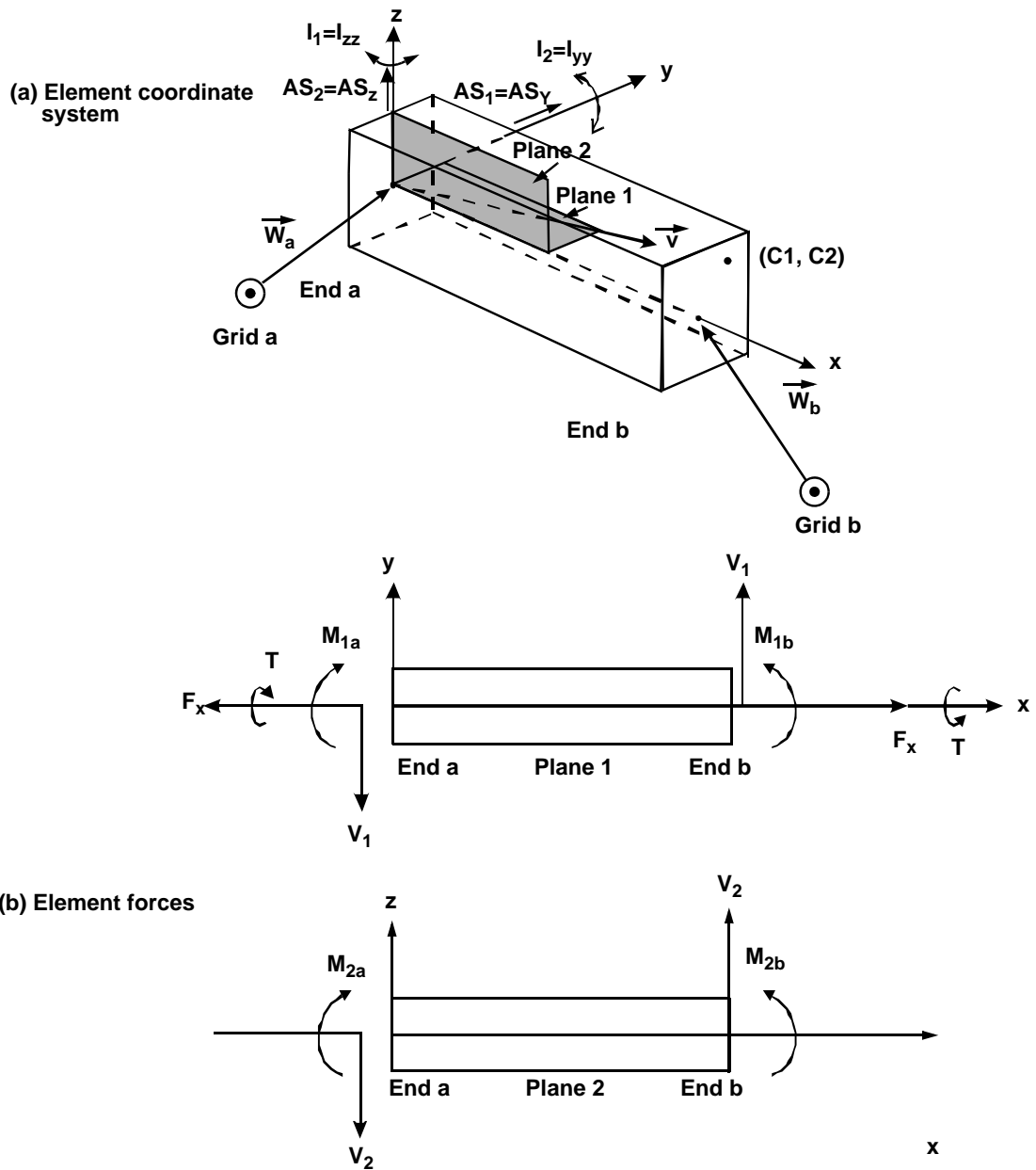


Figure 3-16 Bar

Type 1 - Square

Description: **Figure 3-17** a solid, square element defined by a single design variable, B . Eight stresses are calculated at each end of the element. These include four bending stresses and four shear stresses.

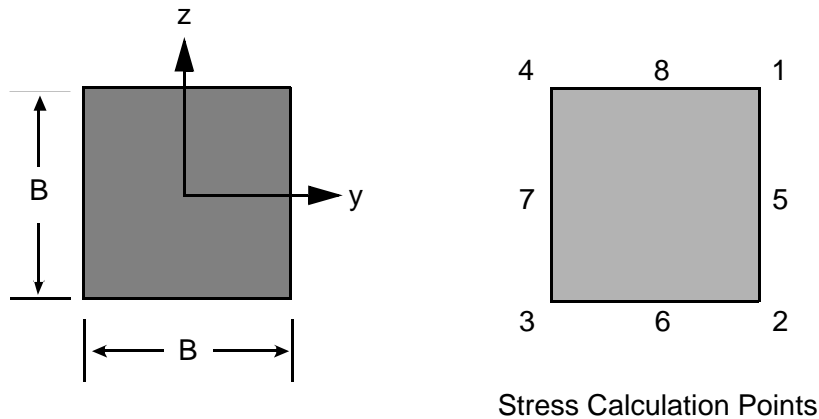


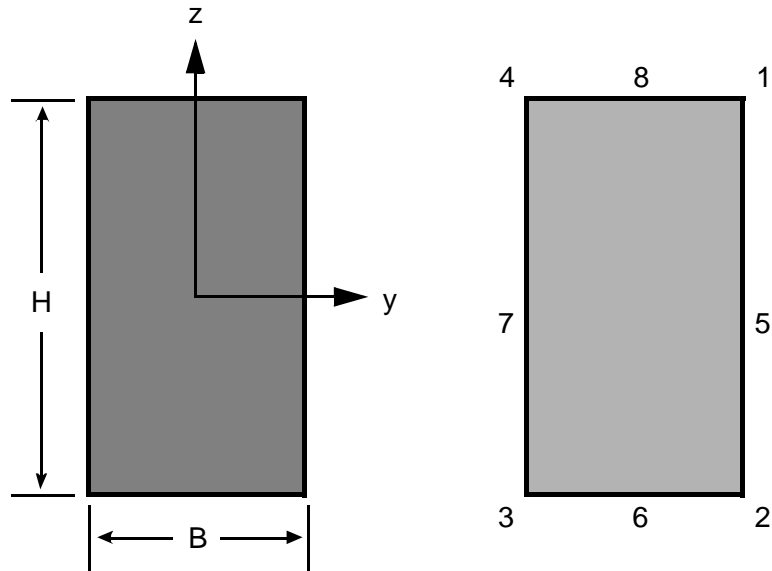
Figure 3-17 Solid, Square Element

Design Variables

Design Variable	Dimension
1	B

Type 2 - Rectangle

Description: Figure 3-18 is a solid, rectangular element defined by design variables, B and H. Eight stresses are calculated at each end of the element. These include four bending stresses and four shear stresses.



Stress Calculation Points

Figure 3-18 Solid, Rectangular Element

Design Variables

Design Variable	Dimension
1	B
2	H

Type 3 - Circle

Description: Figure 3-19 is a solid, round element defined by a single design variable, D. Five stresses are calculated at each end of the element. These include the maximum von Mises (location 1), two bending stresses (locations 2-3) and two shear stresses (locations 4,5).

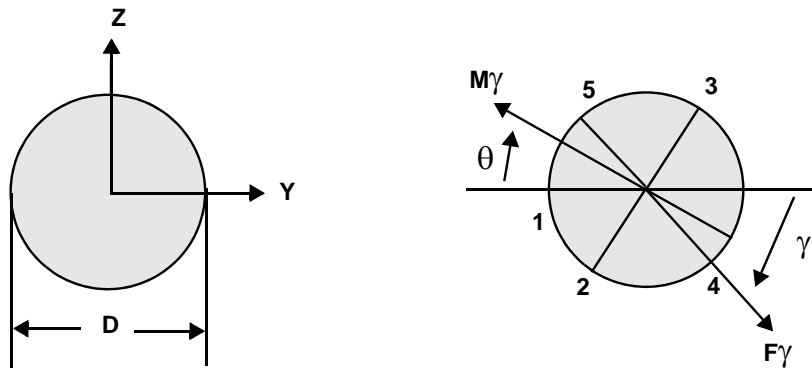


Figure 3-19Solid, Round Element

Design Variables

Design Variable	Dimension
1	D

Type 4 - Tube

Description: Figure 3-20 is a tube (thick or thin) defined by two design variables, D_i and t . Five stresses are calculated at each end of the element, at the maximum bending and shear stress locations. These include three bending stresses (locations 1-3) and two shear stresses (locations 4,5).

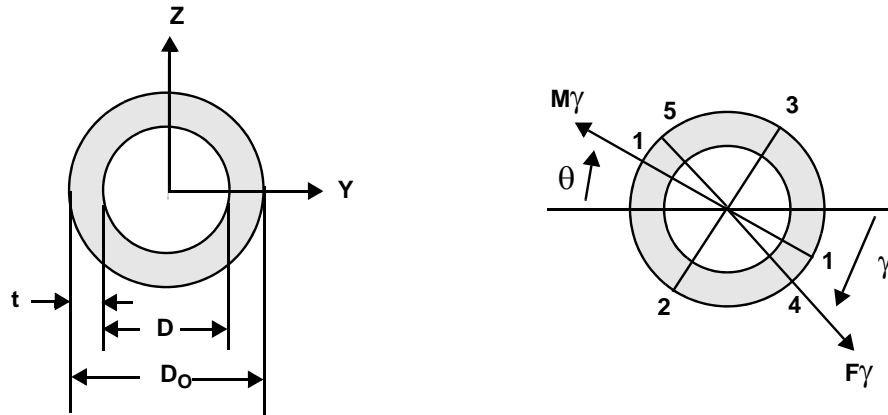


Figure 3-20 Tube

Design Variables

Design Variable	Dimension
1	D_i
2	t

Other Dimensions

$$D_o = D_i + 2t \quad (\text{Eq. 3-37})$$

Type 5 - Spar

Description: Figure 3-21 is a frame element with the cross section described by the sizing variables A_c , H and t . It is intended primarily for use in planar frame structures. Three stresses are computed at each end of the element. These include two bending stresses and one shear stress.

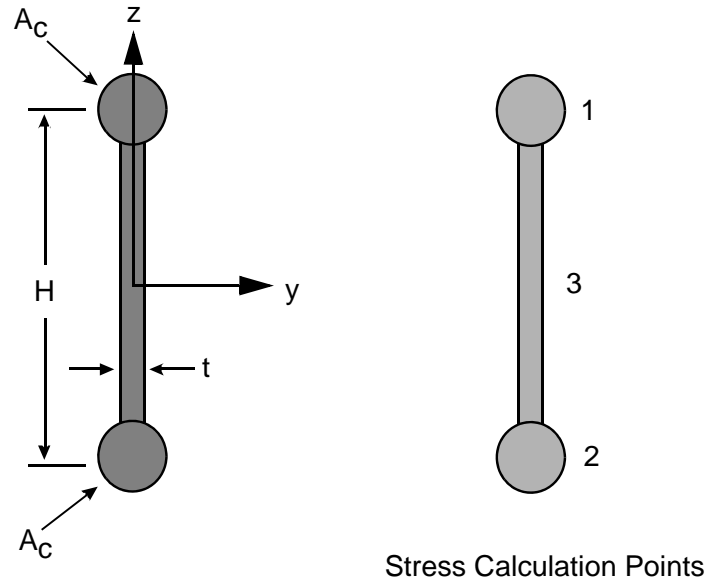


Figure 3-21 Spar

Design Variables

Design Variable	Dimension
1	A_c
2	H
3	t

Type 6 - Box3 (Constant Thickness Box)

Description: Figure 3-22 is a frame element with the cross section described by the sizing variables b , h and t . Eight stresses are computed at each end of the element. These include four bending stresses and four shear stresses.

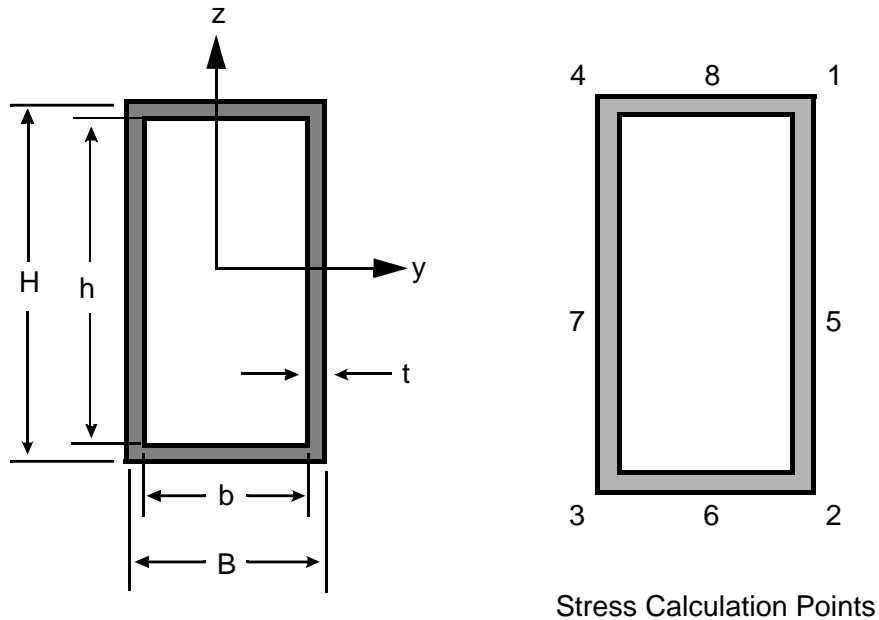


Figure 3-22 Frame Element 1

Design Variables

Design Variable	Dimension
1	b
2	t
3	h

Other Dimensions

$$H = h + 2t \quad (\text{Eq. 3-38})$$

$$B = b + 2t \quad (\text{Eq. 3-39})$$

Type 7 - Box4 (Two Thickness Box)

Description: Figure 3-23 is a frame element with the cross section described by the sizing variables, b , h , t_1 and t_2 . Eight stresses are computed at each end of the element. These include four bending stresses and four shear stresses.

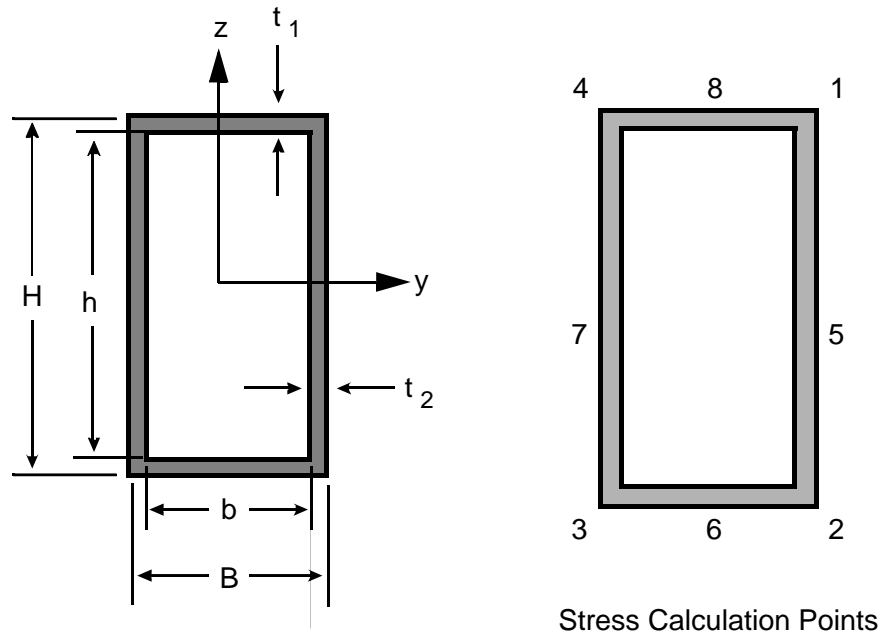


Figure 3-23 Frame Element 2

Design Variables

Design Variable	Dimension
1	b
2	t_1
3	h
4	t_2

Other Dimensions

$$H = h + 2t_1 \quad (\text{Eq. 3-40})$$

$$B = b + 2t_2 \quad (\text{Eq. 3-41})$$

Type 8 - I Beam

Description: Figure 3-24 is a frame element with the cross section described by the sizing variables B , t_1 , h and t_2 . It is intended primarily for use in planar frame structures. Seven stresses are computed at each end of the element. These include four bending stresses and three shear stresses.

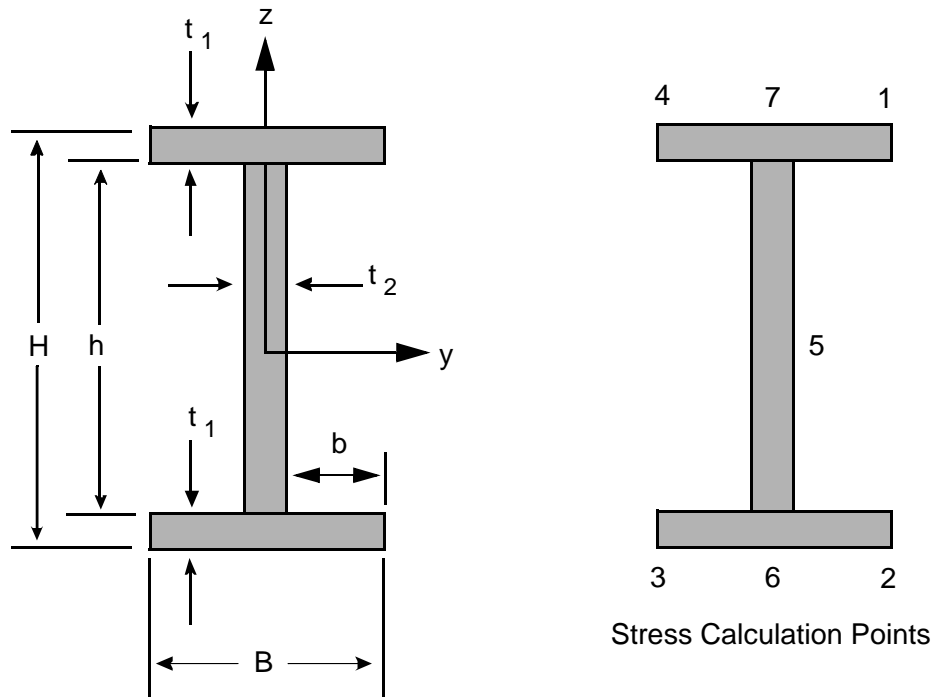


Figure 3-24 Frame Element 3

Design Variables

Design Variable	Dimension
1	b
2	t_1
3	h
4	t_2

Other Dimensions

$$H = h + 2t_1 \quad (\text{Eq. 3-42})$$

$$B = 2b + t_2 \quad (\text{Eq. 3-43})$$

Type 9 - Rail

Description: Figure 3-25t is a frame element with the cross section described by the sizing variables b_1 , t_1 , b_2 , t_2 , h and t_3 . It is intended primarily for use in planar frame structures. Seven stresses are computed at each end of the element. These include four bending stresses and three shear stresses.

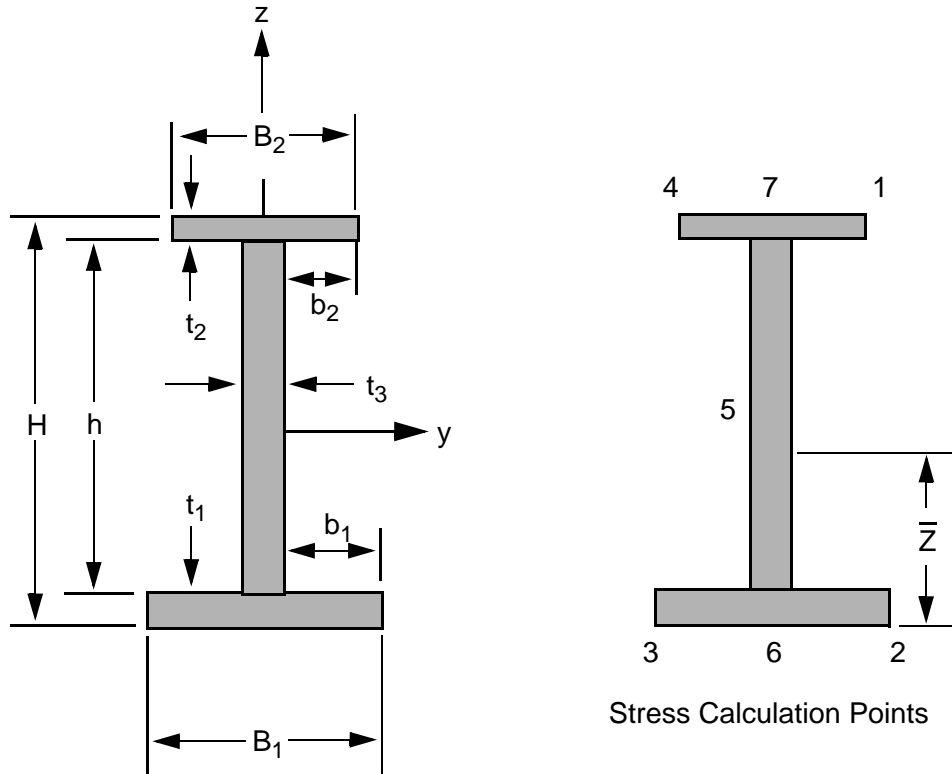


Figure 3-25Rail

Design Variables

Design Variable	Dimension
1	b_1
2	t_1
3	b_2
4	t_2
5	h
6	t_3

Other Dimensions

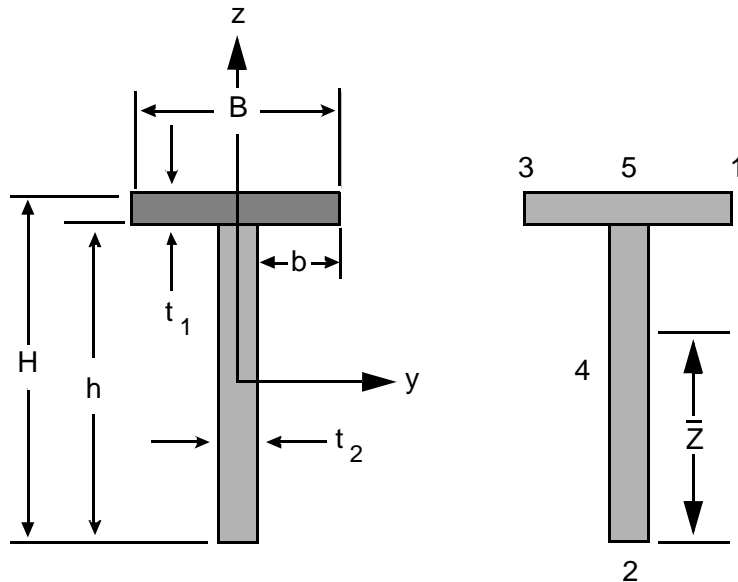
$$H = h + t_1 + t_2 \quad (\text{Eq. 3-44})$$

$$B_1 = 2b_1 + t_3 \quad (\text{Eq. 3-45})$$

$$B_2 = 2b_2 + t_3 \quad (\text{Eq. 3-46})$$

Type 10 - Tee

Description: Figure 3-26, this cross section is described by the sizing variables b , t_1 , h and t_2 . It is intended primarily for use in planar frame structures. Five stresses are computed at each end of the element. These include three bending stresses and two shear stresses.



Stress Calculation Points

Figure 3-26 Tee

Design Variables

Design Variable	Dimension
1	b
2	t_1
3	h
4	t_2

Other Dimensions

$$H = h + t_1 \quad (\text{Eq. 3-47})$$

$$B = 2b + t_2 \quad (\text{Eq. 3-48})$$

Type 11 - Angle

Description: Figure 3-27, this cross section is described by the sizing variables b , t_1 , h and t_2 . Five stresses are computed at each end of the element. These include three bending stresses and two shear stresses. Warping of the section is not accounted for.

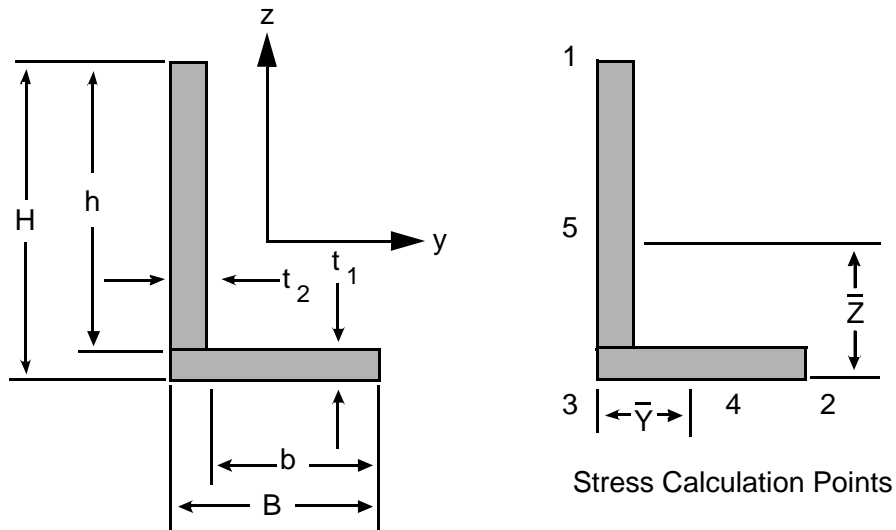


Figure 3-27 Angle

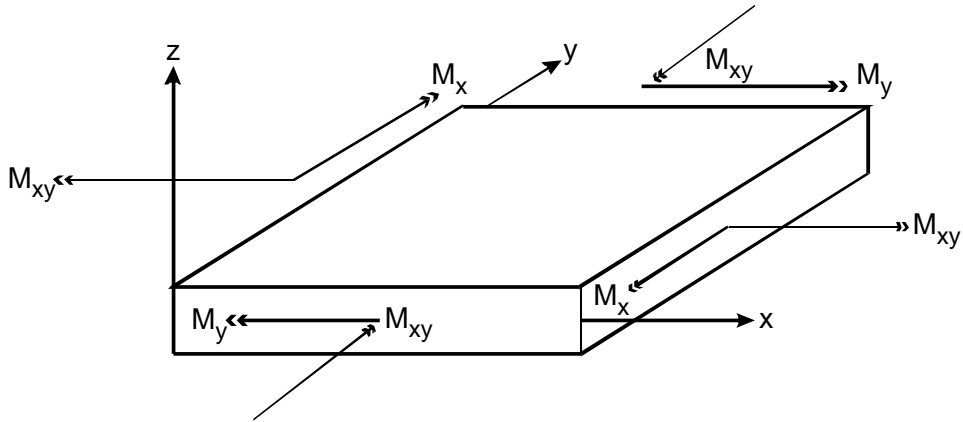
Design Variables

Design Variable	Dimension
1	b
2	t_1
3	h
4	t_2

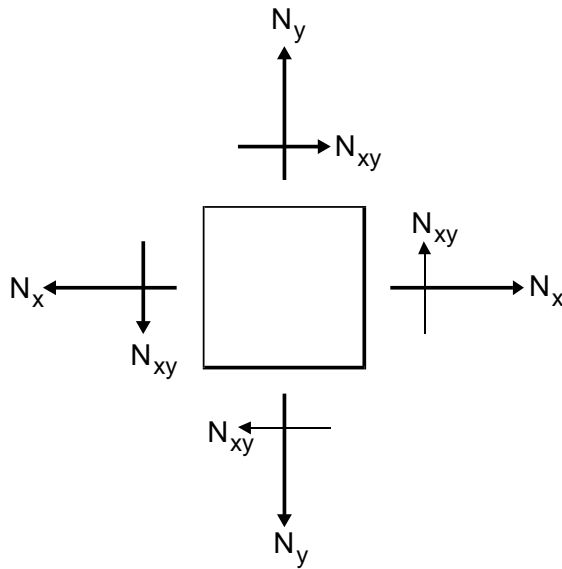
Other Dimensions

$$H = h + t_1 \quad (\text{Eq. 3-49})$$

$$B = b + t_2 \quad (\text{Eq. 3-50})$$



(a) Plate element forces



(b) Membrane element forces

Figure 3-28

Type 12 - Solid (Solid Plate)

Description: Figure 3-29 is defined by design variable t. Stresses are calculated at the center of the top and bottom surfaces of the element.

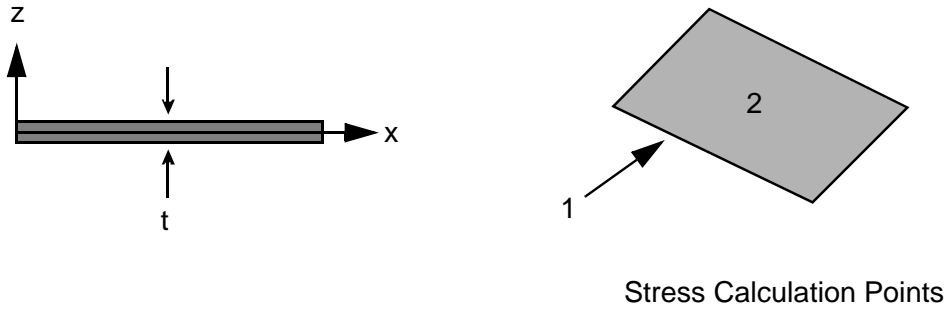


Figure 3-29 Solid Plate Element

Design Variables

Design Variable	Dimension
1	t

Type 13 - Sand (Sandwich Plate)

Description: Figure 3-30 is defined by design variables t and h . Stresses are calculated at the center of the top and bottom surfaces of the element. The core has no bending stiffness.

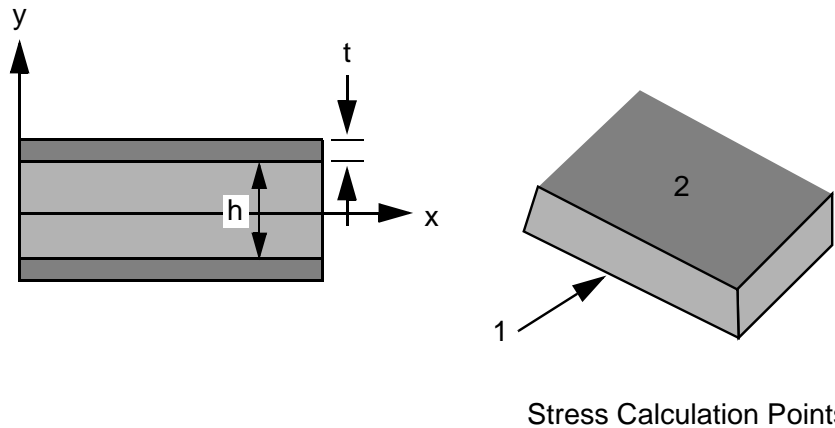


Figure 3-30Sandwich Plate Element

Design Variables

Design Variable	Dimension
1	t
2	h

Type 14: Sand2 (Two Thickness Sandwich Plate)

Description: This is a sandwich plate element defined by design variables t_1 , t_2 and h . Stresses are calculated at the center of the top and bottom surfaces of the element. The core has no bending stiffness. The bending plane is assumed to be in the same plane as corner grids. In other words, bending/membrane coupling does not exist.

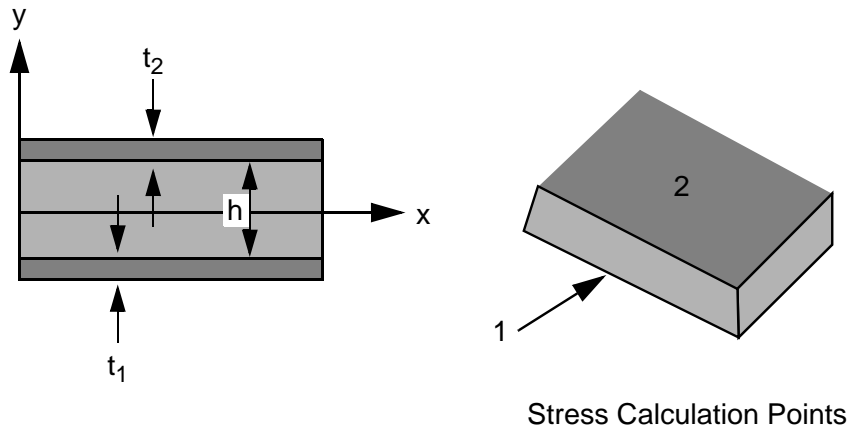


Figure 3-31

Design Variables

Design Variable	Dimension
1	t_1
2	t_2
3	h

3.3.3 Design Variable to Composite Property Relationships (DVPROP4)

The design of composites requires the creation of multiple relationships from design variables to the thicknesses and angles. This can be achieved using multiple DVPROP1 and/or DVPROP2 entries. This standard procedure can be time consuming so an alternative data statement, DVPROP4, is provided in *GENESIS*. This data is intended for the special, but very common, case in which the angles and thicknesses are each linearly linked to a single design variable.

This data does not provide for the design of Z_0 , GE or nonstructural mass of composites. If these properties need to be designed, then DVPROP1 or DVPROP2 entries must be used for those properties.

The basic format of DVPROP 4 is:

1	2	3	4	5	6	7	8	9	10
DVPROP4	ID	PID							
+	LABEL	TC01	TDV1	TCOEF1	AC01	ADV1	ACOE1		
+	LABEL	TC02	TDV2	TCOEF2	AC02	ADV2	ACOE2		
+	LABEL	TC03	TDV3	TCOEF3	AC03	ADV3	ACOE3		
+	etc.								

The ID is a unique DVPROP4 identifier. The PID is the property number of the PCOMP/PCOMPG that is being updated. The LABEL is optional, and is used for describing the layer. The TC0i field is for thickness constant terms. The TDVi are used to specify the thickness design variable IDs. The TCOEFi are to specify the thickness design variable multipliers. The AC0i field is for angle constant terms. The ADVi are used to specify the angle design variable IDs. The ACOEFi are to specify the angle design variable multipliers.

In DVPROP4, continuation line number i corresponds to layer i . The default value of the constant terms is zero while the default value of the multipliers is 1.0.

This data will internally create the following linear relationships:

$$\text{Thickness 1} = \text{TC01} + \text{DVAR}(\text{TDV1}) * \text{TCOEF1}$$

$$\text{Thickness 2} = \text{TC02} + \text{DVAR}(\text{TDV2}) * \text{TCOEF2}$$

$$\text{Thickness 3} = \text{TC03} + \text{DVAR}(\text{TDV3}) * \text{TCOEF3}$$

etc.

$$\text{Angle 1} = \text{AC01} + \text{DVAR}(\text{ADV1}) * \text{ACOE1}$$

$$\text{Angle 2} = \text{AC02} + \text{DVAR}(\text{ADV2}) * \text{ACOE2}$$

$$\text{Angle 3} = \text{AC03} + \text{DVAR}(\text{ADV3}) * \text{ACOE3}$$

etc.

Shape and Sizing Design Capabilities

Example:

Create the following 10 relationships:

Layer 1:

$$\text{Thickness 1} = 2.1 + 2.2 * \text{DVAR 1}$$

$$\text{Angle 1} = 45.0 + 1.1 * \text{DVAR 2}$$

Layer 2:

$$\text{Thickness 2} = 2.1 + 2.2 * \text{DVAR 1}$$

$$\text{Angle 2} = -45.0 - 1.1 * \text{DVAR 2}$$

Layer 3: Not designed

Layer 4:

$$\text{Thickness 4} = \text{DVAR 3}$$

$$\text{Angle 4} = \text{DVAR 4}$$

Layer 5: Thickness 5 = DVAR 5

$$\text{Angle 5} = \text{DVAR 6}$$

Layer 6:

$$\text{Thickness 5} = \text{DVAR 7} = 0.0 + 1.0 * \text{DVAR 7}$$

$$\text{Angle 5} = \text{DVAR 8} = 0.0 + 1.0 * \text{DVAR 8}$$

	1	2	3	4	5	6	7	8	9	10
DVPROP4	1001	100								
+	layer1	2.1	1	2.2	45.0	2	1.1			
+	layer2	2.1	1	2.2	-45.0	2	-1.1			
+	layer3									
+	layer4		3			4				
+	layer5		5			6				
+	layer6	0.0	7	1.0	0.0	8	1.0			

This data is completely equivalent to the following 10 DVPROP1 data statements:

	1	2	3	4	5	6	7	8	9	10
DVPROP1	1001	100	101	2.1						
+	1	2.2								
DVPROP1	1002	100	501	45.0						
+	2	1.1								
DVPROP1	1003	100	102	2.1						
+	1	2.2								
DVPROP1	1004	100	502	-45.0						
+	2	-1.1								
DVPROP1	1005	100	104							
+	3									
DVPROP1	1006	100	504							
+	4									
DVPROP1	1007	100	105							
+	5									
DVPROP1	1008	100	505							
+	6									
DVPROP1	1009	100	106							
+	7									
DVPROP1	1010	100	506							
+	8									

GENESIS automatically converts DVPROP4 entries into equivalent DVPROP1 data. The ECHO=SORT command will print the internally created DVPROP1, instead of the user input DVPROP4.

3.3.4 General Nonlinear Response Generation (DRESP2)

The **DRESP2** input data can be used to construct user defined responses that can be used as the objective function or can be constrained. These responses can be nonlinear functions of design variables, nodal locations, tabled constants, and structural responses from different load cases. These user defined responses can be used to construct element buckling constraints and constraints on the shape of a structure, as examples. The shape constraints are used to keep the design realistic and practical. Many other types of responses can be generated. Since the DRESP2 capability is so flexible and powerful its use is only limited by the imagination of the user. The format of the DRESP2 data is:

	1	2	3	4	5	6	7	8	9	10
DRESP2	ID	LABEL	EQID	REGION						
+	"DVAR"	NDV1	NDV2	NDV3	NDV4	NDV5	NDV6	NDV7	NDV8	
+	Blank	NDV9	NDV10	...						
+	"DTABLE"	NC1	NC2	...						
+	"DGRID"	NG1	NGC1	NG2	NGC2	...				
+	"DRESP1"	NR1	NR2	...						

The alternate format is

	1	2	3	4	5	6	7	8	9	10
DRESP2	ID	LABEL	EQID	Blank						
+	"DVAR"	NDV1	NDV2	NDV3	NDV4	NDV5	NDV6	NDV7	NDV8	
+	Blank	NDV9	NDV10	...						
+	"DTABLE"	NC1	NC2	...						
+	"DGRID"	NG1	NGC1	NG2	NGC2	...				
+	"DRESP1L"	NR1	LIDR1	NR2	LIDR2	NR3	LIDR3	...		

The keywords DVAR, DTABLE, DGRID, and DRESP1 or DRESP1L are used to indicate that the following data is design variable ID's, tabled constant position number, grid ID's and component numbers, or DRESP1 and DRESPG ID's and load cases, respectively. More than eight ID's can be input using continuation lines (as shown for the DVAR ID's above). Not all four types of ID's must be referenced.

For example if the response is only a function of grid point locations the data would be:

	1	2	3	4	5	6	7	8	9	10
DRESP2	ID	LABEL	EQID							
+	"DGRID"	NG1	NGC1	NG2	NGC2	...				

However, if more than one keyword is used, they must appear in the order shown above.

The DRESP2 data references DEQATN data which contains the equation that is used to calculate the response. The design variable values, tabled constant values, grid point locations, and structural response values are used as the variables in the equation.

As an example, consider the buckling of a pinned end round ROD element. The critical buckling load is

$$F_{CR} = -\frac{\pi EA^2}{4L^2} \quad (\text{Eq. 3-51})$$

Buckling of the ROD element can be avoided by using the following constraint function:

$$\frac{4FL^2}{\pi EA^2} \geq -1 \quad (\text{Eq. 3-52})$$

Note that the constraint is a function of a structural response; the force in the ROD, and a design variable value; the area of the ROD. Assuming $E=1.0E6$ and $L=100$ the data needed to generate this constraint would be:

	1	2	3	4	5	6	7	8	9	10
DVAR	1	AREA	3.0	1.0	10.0					
DRESP1	40	RODFA	FORCE	ELEM		1	1			
DRESP2	10	RBUCK	100							
+	DVAR	1								
+	DRESP1	40								
DEQATN	100	B(A,F) = 4.*F*100.**2/(3.14159*1.0E6*A**2)								
DCONS	10	1	-1.0	1.0E30						

DRESP1 40 specifies the force at end A of ROD 1. DVAR 1 controls the area of this ROD element. DRESP2 10 references DEQATN 100 which contains the equation for the buckling response function. This response is constrained to be greater than -1.0 by the DCONS data.

Shape and Sizing Design Capabilities

The values of π and E could be tabled constants. In this case the data would be:

	1	2	3	4	5	6	7	8	9	10
DVAR	1	AREA	3.0	1.0	10.0					
DTABLE	PI	3.141593	YOUNGS	1.0E6						
DRESP1	40	RODFA	FORCE	ELEM		1	1			
DRESP2	10	RBUCK	100							
+	DVAR	1								
+	DRESP1	40								
DEQATN	100	$B(A,F) = 4.*F*100.**2/(PI*YOUNGS*A**2)$								
DCONS	10	1	-1.0	1.0E30						

Finally, suppose the length of the ROD element is not constant and is equal to the distance between the X coordinates of grid points 2 and 3. The data for this case would be:

	1	2	3	4	5	6	7	8	9	10
DVAR	1	AREA	3.0	1.0	10.0					
DTABLE	PI	3.141593	YOUNGS	1.0E6						
DRESP1	40	RODFA	FORCE	ELEM		1	1			
DRESP2	10	RBUCK	100							
+	DVAR	1								
+	DGRID	2	1	3	1					
+	DRESP1	40								
DEQATN	100	$B(A,X2,X3,F) = 4.*F*(X2-X3)**2/(PI*YOUNGS*A**2)$								
DCONS	10	1	-1.0	1.0E30						

If the DRESP1 keyword is used in the DRESP2 data then the structural responses are all taken from the LOADCASE specified on the DCONS or DOBJ data for this DRESP2. Since all the responses are from the same LOADCASE, static, frequency, dynamics and heat transfer responses cannot be mixed.

Use of the DRESP1L keyword in the DRESP2 data allows responses from different LOADCASEs to be mixed. When the DRESP1L keyword is used a specific LOADCASE is listed after each DRESP1 ID. Since the LOADCASEs are specified in DRESP2 data, the LID field on the DCONS data must be left blank. The LOADCASE specified on the DOBJ data must be the same as the first LOADCASE ID in the DRESP1L data (LIDR1). Also note that the REGION field on the DRESP2 must be left blank when DRESP1L data is used. As an example, consider a response that is the sum of the X direction displacement of GRID 10 in LOADCASE 3 and the first frequency in LOADCASE 6. The input data for this response would be:

	1	2	3	4	5	6	7	8	9	10
DRESP1	90	DISP1	DISP				1	10		
DRESP1	95	MODE1	FREQ				1			
DRESP2	200	DISPFRQ	990							
+	DRESP1L	90	3	95	6					
DEQATN	990	F(A,B)=A+B								
DCONS	200	34	55							

More examples of DRESP2 data are given in the discussion of the equation utility in the [Equation Utility](#) (p. 271).

3.3.5 Constraint Screening (DSCREEN)

In many structural design problems there are hundreds and even thousands of design constraints. Most of these constraints are on element responses (force, stress, and strain). Consider a plate structure modeled with 3000 finite elements and subjected to five different static loadings. If the von Mises stresses on the bottom and top surfaces of every element are constrained to be less than the yield stress in each load case a total of 30,000 stress constraints are needed.

Although there are thousands of constraints it is likely that only a small number of them are important to the design process during each design cycle. Since the cost of the sensitivity analysis is proportional to the number of constraints, it is desirable to only deal with the constraints that are critical (at or outside the constraint bound) and potentially critical (near the constraint bound). Constraint screening is a procedure used to eliminate the constraints that are not currently important to the design process. Constraint screening is performed for each design cycle as different sets of constraints will become critical as the design changes. In the output, constraint information is only printed for retained constraints.

Constraint screening is performed for each type of response: stress, strain, force, displacement, velocity, acceleration, temperature, mass, volume, strain energy, frequency constraint, buckling load factors and equations. There are two parameters for each constraint type that control the screening procedure. The first is called the constraint truncation threshold (TRS). It is used to define the value above which constraints are considered potentially critical. The default value is -0.5 which means that constraints that are within 50% of their critical value are considered potentially active. If the design does not change much from design cycle to design cycle this value can be raised to say -0.2 (constraints must be within 20% of their critical value in order to be retained). If it is observed that constraints which are not retained in one design cycle are violated in the next design cycle then the value of TRS should be lowered to say -0.7 (constraints within 70% of their critical value are retained). The default value of -0.5 is quite conservative and will usually prevent nonretained constraints from becoming violated.

The second parameter that controls the constraint screening process is used to specify the maximum number of constraints to be retained in a specific region of the structure (NSTR). This is used to limit the number of constraints that are retained in highly stressed areas of the structure. For example, consider a uniform thickness plate that contains a stress concentration. Many elements around the stress concentration may be highly stressed and therefore retained. However, it is likely that if the stress is reduced in the most highly stressed elements, the stress in the other elements will also be reduced. Therefore, it is advantageous to only retain the most highly stressed elements.

The default screening parameter values can be overridden using the **DSCREEN** input data. The format for this data is:

1	2	3	4	5	6	7	8	9	10
DSCREEN	TYPE	TRS	NSTR						

where TYPE is either DISP, TEMP, STRESS, STRAIN, FORCE, DRESPG, MASS, VOLUME, FREQ, LAMA, INERTIA, SENERGY and EQUA which is used to specify the screening parameters for the constraints on DRESP2 and DRESP3 responses.

For frequencies, it is recommended that the bound value that should not be included in optimization be left blank or set to $\pm 1.0E+30$ so that no constraint is generated.

The different regions of the structure are defined on the DRESP1 and DRESP2 input data. The default is to put all the elements associated with a property entry in a single region. All of the elements associated with a DRESP1 entry are placed in the same region if a region identifier is given. All of the elements associated with DRESP1 entries with the same region identifier are placed in the same region. It is recommended that the region identifier be left blank (the default).

Screening Control for Frequency Response

For frequency dependent responses that come from frequency response analysis a DOPT parameter, **FREQREG**, can be used to further control the number of retained constraints.

FREQREG defines how responses for different loading frequencies in a frequency response loadcase are grouped together in screening regions.

If $FREQREG > 0$ then FREQREG corresponds to the maximum number of dynamic loading frequencies per region.

If $FREQREG < 0$ then $ABS(FREQREG)$ corresponds to the maximum number of regions per frequency response loadcase.

The least aggressive screening method corresponds to $FREQREG=1$, which puts every loading frequency into a different region. The most aggressive screening method correspond to -1, which puts all loading frequencies within a loadcase into the same screening region.

An aggressive choice can cause the program to retain fewer responses, which will reduce sensitivity calculation time. However, an aggressive choice runs the risk of having important responses screened out, leading to an increased number of design cycles before convergence.

Example 1: Conservative screening:

	1	2	3	4	5	6	7	8	9	10
DOPT										
+	FREQREG	1								

or :

	1	2	3	4	5	6	7	8	9	10
DOPT										
+	FREQREG	-500								

The two entries above are equivalent if there are no more than 500 loading frequencies in a loadcase.

Example 2: Aggressive screening:

	1	2	3	4	5	6	7	8	9	10
DOPT										
+	FREQREG	500								

or :

	1	2	3	4	5	6	7	8	9	10
DOPT										
+	FREQREG	-1								

The two entries above are equivalent if there are no more than 500 loading frequencies in a loadcase.

Example 3: Intermediate screening:

	1	2	3	4	5	6	7	8	9	10
DOPT										
+	FREQREG	50								

or :

	1	2	3	4	5	6	7	8	9	10
DOPT										
+	FREQREG	-10								

The two entries above are equivalent if there are exactly 500 loading frequencies in a loadcase.

3.3.6 Move Limits

GENESIS uses the approximation concepts approach to structural optimization. In this approach the structural responses are approximated using first order Taylor Series expansions. While these approximations are very accurate near the design point, they tend to degrade as the design is changed during the design cycle. For this reason, move limits are placed on the design variables and analysis model properties during each design cycle. Move limits also keep the design from changing so much that nonretained constraints become violated during the design cycle. Move limits on the locations of grid points are also used to protect the accuracy of the approximations and to keep nonretained constraints from becoming highly violated.

Move limits for the design variables are specified on the DVAR data. The format for the DVAR data is:

1	2	3	4	5	6	7	8	9	10
DVAR	ID	LABEL	INIT	LB	UB	DELX	DXMIN		

The value of DELX is the fractional value that the design variable is allowed to change during each design cycle. The default value for DELX is 0.5. This means that the design variable is allowed to change by a maximum of 50% during each design cycle. In some cases the design variables become very small. In this case move limits based on a fractional change may be too restrictive. For example if the design variable value is 0.001 then the allowable change is only ± 0.0005 . To overcome this problem a minimum move limit is used. The minimum move limit is specified by DXMIN. If the fractional allowable change in the design variable is less than DXMIN then DXMIN is used for the move limit. If DXMIN is 0.1 then the move limits in the above example would be ± 0.1 . The default value for DXMIN is 0.1 if the absolute value of the original design variable is less than 1.0, or 10% of the absolute value of the original design variable value if its absolute value is greater than 1.0. If the design variable move limits allow the design variable to move outside of the bounds on the design variable then the move limit is reset to be the bound value. The algorithm used to determine the move limits for a design cycle is shown below.

$$\Delta_{\min} = \begin{cases} \text{DXMIN} & \text{if } |X_{\text{Init}}| \leq 1.0 \\ \text{DXMIN} * |X_{\text{Init}}| & \text{if } |X_{\text{Init}}| > 1.0 \end{cases} \quad (\text{Eq. 3-53})$$

$$\Delta = \text{MAX}(\text{DELX} * X, \Delta_{\min}) \quad (\text{Eq. 3-54})$$

$$X^L = \text{MAX}(X - \Delta, X^{LB}) \quad (\text{Eq. 3-55})$$

$$X^U = \text{MIN}(X + \Delta, X^{UB}) \quad (\text{Eq. 3-56})$$

Shape and Sizing Design Capabilities

where X is the design variable value. The default values of DELX and DXMIN are recommended. If the responses are very nonlinear functions of the design variables or very sensitive to small changes in the design variables then smaller move limits should be used. The move limits are automatically adjusted during the design process to protect the accuracy of the approximations and speed convergence.

Move limits are also placed on the analysis model properties. They are specified on the DVPROP data as follows:

1	2	3	4	5	6	7	8	9	10
DVPROP1	ID	PID	FID	C0	PMIN	DELP	DPMIN		
+	DVID1	C1	DVID2	C2	...				
\$									
DVPROP2	ID	PID	FID	EQID	PMIN	DELP	DPMIN		
+	"DVAR"	DVID1	DVID2	DVID3	...				
+	Blank	DVIDi	...						
+	"DTABLE"	CID1	CID2	CID3	...				
+	Blank	CIDi	...						
\$									
DVPROP3	ID	PID	ELTYPE	ISHEAR	PMIN	DELP	DPMIN		
+	CSD1	CSD2	CSD3	CSD4	...				

PMIN is the minimum allowable value of the property. The default value for PMIN is 1.0E-10 except for properties that can be negative, such as stress recovery points and I_{12} , in which case it is -1.0E35. DELP is the allowable fractional change in the property during each design stage. DPMIN is the minimum move limit value. The property move limits are set in using the same procedure that is used for the design variable move limits:

$$\Delta_{\min} = \begin{cases} \text{DPMIN} & \text{if } |P_{\text{Init}}| \leq 1.0 \\ \text{DPMIN} * |P_{\text{Init}}| & \text{if } |P_{\text{Init}}| > 1.0 \end{cases} \quad (\text{Eq. 3-57})$$

$$\Delta = \text{MAX}(\text{DELP} * P, \Delta_{\min}) \quad (\text{Eq. 3-58})$$

$$P^L = \text{MAX}(P - \Delta, \text{PMIN}) \quad (\text{Eq. 3-59})$$

$$P^U = P + \Delta \quad (\text{Eq. 3-60})$$

The default values for DELP and DPMIN are 0.5 and 0.1 respectively.

Shape and Sizing Design Capabilities

Move limits can also be placed on grid point locations in shape design problems. The move limits are specified on the GRID data continuation line which has the format:

1	2	3	4	5	6	7	8	9	10
GRID	ID	CP	X1	X2	X3	CD	PS		
+	MV	X1L	X1U	X2L	X2U	X3L	X3U	XR	

MV is a switch that is used to specify whether the limits are move limits for each design cycle (MV=1) or bounds on the locations of the grid points (MV=0, the default).

If MV=1 then the XiL and XiU specify how much the grid point location can change during the design cycle. The algorithm that determines the grid point move limits in this case is:

$$XYZ^L = XYZ + X_iL \quad (\text{Eq. 3-61})$$

$$XYZ^U = XYZ + X_iU \quad (\text{Eq. 3-62})$$

Note that XiL must be a negative number. This type of move limit is usually used on grid points near stress concentrations because small changes in the shape of a structure near a stress concentration can cause large changes in the stress field in this area. The XR data is a resultant (spherical) move limit and therefore must always be positive. It causes the location of the grid point at the end of the design cycle to be within a sphere of radius XR centered at the location of the grid point at the beginning of the design cycle.

If MV=0 then the XiL and XiU specify bounds on the location of the grid point throughout the design process. The algorithm that determines the grid point move limits in this case is:

$$XYZ^L = X_iL \quad (\text{Eq. 3-63})$$

$$XYZ^U = X_iU \quad (\text{Eq. 3-64})$$

XR is not used if MV=0. This type of move limit is normally used to prevent the structure from interfering with neighboring structures.

3.3.7 Optimization Process Control (DOPT)

The user is able to override the default optimization parameters using the **DOPT** input data. The maximum number of design cycles, convergence parameters, move limits, and other parameters can be changed using this data.

Maximum Number of Design Cycles

The maximum allowable number of design cycles (DESMAX) has a default value of 10. If the initial design is far from the optimum design this value should be raised to 15 or 20. It should also be raised if the responses are very nonlinear functions of the design variables. It is observed that between 5 and 10 design cycles are needed for most practical design problems. If the analysis is very expensive, it is advisable to set DESMAX to a smaller number. The design process can always be continued using the *GENESIS* restart capability. The restart capability is explained in **Restart Capability** (p. 321).

Minimum Number of Design Cycles

The minimum number of design cycles (**DESMIN**) has a default value of 0. A value n , greater than zero, can be used to force the program to run at least n approximate optimization design cycles before stopping.

Move Limits Parameters

The default move limit parameters for the design variables (**DELX** and **DXMIN**) and properties (**DELP** and **DPMIN**) . See **Move Limits** (p. 135) for more information.

Also the **DXFRAC** move limit parameter used for topometry design variable can be reset on the DOPT input data.

Hard Convergence Parameters

There are two conditions that must be met for hard convergence. The first is that all the design constraints must not be violated by more than a small value (**GMAX**). The default value for the allowable constraint violation is 0.005 or 0.5%. The other condition is that either the relative change in the objective function for two consecutive design cycles be less than **CONV1** (default = 0.001 or 0.1%) or the absolute change in the objective function for two consecutive design cycles be less than **CONV2** (default = 0.001 or 0.000001 when DMATCH is used). CONV2 is defined as $\text{MAX}(\text{CONV2} * |\text{OBJ}^0|, 1.0\text{E-}19)$. This means that it is the maximum of 0.1% of the initial objective function value or 1.0E-19. The values of GMAX, CONV1, and CONV2 can be changed on the DOPT data. If the user wishes to get very close to the optimum

design, the values of CONV1 and CONV2 can be made smaller. Larger values of CONV1 and CONV2 will lead to faster convergence but the final design may not be optimum. The maximum allowable constraint violation can also be reduced or increased. It is not recommended that GMAX be made smaller than 0.003.

Soft Convergence Parameters

There are up to four conditions that must be met for soft convergence. First, the maximum relative design variable change must be less than **CONVDV**. Second, the maximum relative change in the analysis model properties must be less than **CONVPR**. Third, the maximum absolute change in the location of the shape design grid points in the input coordinate system must be less than **CONVLC**. Last, the change in the maximum constraint value must be less than **CONVCN**. As with the hard convergence parameters, smaller values may lead to a more optimal solution and more design cycles while larger values will lead to less optimal solutions but with fewer design cycles. The default values for CONVDV, CONVPR, CONVLC and CONVCN are 0.001. However, when DMATCH is used, the default value for CONVDV is 0.000001. These default values can be changed separately using the DOPT input data.

Basis/Perturbation Vector Default Switch

The optimization parameter **BASIS** is used to tell the program the default on field 9 of DVGRID and DVGRIDC data. For basis vectors the parameter should be 1, for grid perturbation vectors the parameter should be 0. This parameter has no default, and in most cases it has to be explicitly specified if DVGRID, DVGRIDC and/or DVSHAPE data is used.

Tolerance/Switches Parameters

The parameters **DVTOL** and **PTOL** are used for checking the input data. If the calculated value of a dependent design variable is different from the user input initial value by more than DVTOL then a fatal error message is issued and the program terminates. If the calculated value of an analysis model property is different from the user input value by more than PTOL then a fatal error message is issued and the program terminates. The default value for both DVTOL and PTOL is 1.0E35.

The switches **SGENEL**, **SK2UU** and **SM2UU** are used to control the checking of grids referenced by GENEL, K2UU and M2UU elements. Because these elements are not designable, their grids should not be referenced by DVGRID data. A value of 0 (zero), the default, in these parameters will cause the program to stop if any of their grids are being designed. A value of 1 will allow the program to continue even if grids connected to these elements are designed.

The parameters **RPERT1** and **RPERT2** are used for avoiding using small perturbation (typically created by third party software). **RPERT1** Controls the relative value of the cut off perturbation. If **RPERT1** = 0.0, the program will ignore this cut-off. **RPERT2** Controls the absolute value of the cut-off perturbation. If **RPERT2** = 0.0, the program will ignore this cut-off value.

Method Parameters

The switch **IUGRAD** is used to specify how the gradients for responses generated by a separate user program are calculated. See **Using External Analysis Programs** (p. 296) for a discussion of user program responses and their gradient calculations.

The switch **ADJOIN** is used to control the method used to calculate sensitivities of static responses. A value of 1 will cause *GENESIS* to use the adjoint method. A value of 0 will cause *GENESIS* to use the direct method. The default is -1, which causes *GENESIS* to automatically select the method that has the best performance.

The parameter **OPTGRID** controls how GRID data is written in the pname.OPT file. A value of 1, the default, causes *GENESIS* to write each GRID data in the same input coordinate system (CP) that was used in the input file. A value of 0 causes the GRID data to be written in the basic coordinate system.

Design Variable Reset

The parameter **DVINIT** allows the initial value of the design variables to be set to their upper or lower bounds. A value of -1 will cause *GENESIS* to reset the initial value of all DVARS to their lower bounds. A value of 1 will cause *GENESIS* to reset the initial value of all DVARS to their upper bounds. A value of 0, the default, will leave the initial values unchanged.

Optimizer Selection

The parameter **OPTM** controls which optimizer is used to solve the approximate optimization problems. A value of 0, the default, selects the DOT optimizer. A value of 1 selects the BIGDOT optimizer.

Stress Ratio Parameters

The **ISMET** parameter is used to control the stress ratio resizing method. If **ISMET**=0, *GENESIS* will not use stress ratio (this is the default). If **ISMET** = 1 or 3, *GENESIS* will not change the design variables that reference elements that do not have stress ratio constraints. The difference between 1 and 3 is that in the first case *GENESIS* would stop due to hard convergence. Between 1 and 3, 3 is recommended for most cases. Option 1 should be picked over 3 only when analysis is very time consuming. If **ISMET**=2 or 4, *GENESIS* will attempt to change the dvar associated to all stress ratio designable elements. The difference between 2 and 4 is that in the first

case *GENESIS* would stop due to hard convergence. Between 2 and 4, 4 is recommended for most cases. Option 2 should be pick over 4 only when analysis is very time consuming. The number of stress ratio design cycles is controlled by the **ISRMAX** parameter (default is 3).

DOT Optimizer Parameters

The parameter **ITRMOP** controls the number of consecutive iterations DOT must satisfy the absolute or relative convergence criteria before the approximate optimization problem is terminated. Usually ITRMOP should be at least 2 because it is common to make little progress in one iteration, only to make major progress on the next. Therefore, the default ITRMOP = 2 will allow a second try before terminating. If progress toward the optimum seems slow, but consistent, and function evaluations are not too expensive, it may improve the solution to increase ITRMOP to a value from 3 to 10.

The parameter **ITMAX** sets the maximum number of iterations in the DOT optimizer for each design cycle. The default is 100.

The switch **IPRINT** is used to control the amount of printed output generated by the optimization program DOT. It is included for debugging purposes and in most cases should be set to the default value of 0.

Contact Analysis Optimization Parameters

The parameter **CNTDC** controls the frequency at which full contact analysis is performed. The default value of 1 causes the program to perform full contact analysis in every design cycle. A value n, greater than one, causes the program to perform full contact analysis every n-th design cycle. The program, however, will always perform full contact analysis in the first and last design cycles. For the design cycles where no full contact analysis is performed, the parameter **CNTIT** provide a reduced maximum number of contact analysis iterations.

3.3.8 Automatic Generation of Element Stress Constraints

GENESIS has the capability to automatically generate stress constraints for all elements in every static LOADCASE and LOADCOM. This leads to a large reduction in design input data preparation. When this option is requested, stress constraints are generated automatically using the stress limits from the material input data. For example, the upper bound stress constraint for a ROD element would be the value from the stress limit for tension, ST, data listed on the MAT1 continuation line.

Automatic generation of stress constraints is requested using the **DCONS** input data. The format of this data used to request this feature is:

1	2	3	4	5	6	7	8	9	10
DCONS	STRESS								

This command automatically generates DRESP1 data with the following characteristics:

1. The RID is always 99999.
2. The label is “AUTOMATC”.
3. Stress constraints are generated in a property by property, as opposed to element by element, basis.
4. Each DRESP1 data is in a separate region.

DCONS data is generated for each DRESP1 with the same RID and “ALL” static LOADCASEs and LOADCOMs are flagged. The upper and lower bounds are determined from the material limits on the MATi data referenced by each property. No stress constraints are generated for materials without stress limits.

Element Stress Constraints

Stress constraints are automatically generated for all elements except CELAS1/2 elements, composite elements that do not specify a failure theory and solid elements that reference MAT9 materials. The procedure used to automatically generate stress constraints is slightly different for each property type.

ROD elements

The stress at end A of each ROD element is constrained to be greater than the stress limit in compression (SC) and less than the stress limit in tension (ST) listed on the MAT1 input data. If GRAVity or CENTrifugal loads are present, then the stress at end B of each ROD element is also constrained because it may be different than that at end A. Note that the stress limit in compression (SC) must be a negative number.

BAR elements

The stress recovery procedure for BAR elements from the design element library (DVPROP3) is different than that of the generic BAR elements.

Generic BAR elements

Upper and lower bound stress constraints are generated for all eight stress recovery points on the generic BAR elements. The stresses are constrained to be greater than stress limit in compression (SC), a negative number, and less than the stress limit in tension (ST) listed on the MAT1 input data.

Element library BAR elements

Upper and lower bound stress constraints are generated for each normal stress on every library BAR element. These stresses are constrained to be greater than stress limit in compression (SC), a negative number, and less than the stress limit in tension (ST) listed on the MAT1 input data. If shear stresses are calculated for the element, then upper and lower bound stress constraints are generated for each shear stress. These stresses are constrained to be greater than the negative of the shear stress limit (SS) and less than the shear stress limit listed on the MAT1 input data. Note that stress constraints are automatically generated only for the library elements supplied by VR&D. If the user adds extra elements to the library and desires constraints on the stresses in these elements, the constraints must be generated through DRESP1 and DCONS data entries.

Plate/Shell elements

An upper bound von Mises stress constraint is generated for both surfaces of all of the TRIA3, TRIA6, QUAD4 and QUAD8 elements. The von Mises stresses are constrained to be below the von Mises stress limit (SS) on the MAT1, MAT2, and MAT8 material data entries referenced by MID1 in the PSHELL data. If MID1 is blank, then the material data referenced by MID2 will be used. Note that if both plate/shell elements and BAR elements from the element library with shear stresses or composite elements reference the same MAT1 data erroneous constraints will be generated. This is because the two-dimensional elements require the von Mises stress limit while the BAR elements require the maximum stress limit. In this case two different MAT1 entries must be used.

Shear elements

An upper bound shear stress constraint will be generated for the largest (in magnitude) shear stress in all SHEAR elements. The maximum shear stress is constrained to be below the shear stress limit (SS) on the MAT1 data.

Composite plate elements

The failure index for each layer of the composite is automatically constrained to be less than or equal to 1.0. For automatic stress constraints to be generated for composite elements, a failure theory must be specified in the PCOMP/PCOMPG data, and stress limits must be specified for the material of every layer.

Axisymmetric elements

An upper bound von Mises stress constraint is generated for the von Mises stress at the centroid of axisymmetric elements. The von Mises stress is constrained to be below the von Mises stress limit (SS) on the MAT1 material data entries. If an axisymmetric element references a MAT3 material, no automatic stress constraints will be generated. Note that if both axisymmetric elements and BAR elements from the element library with shear stresses reference the same MAT1 data, erroneous constraints will be generated. This is because the axisymmetric elements require the von Mises shear stress limit while the BAR elements require the maximum shear stress limit. In this case, two different MAT1 entries must be used.

Three-dimensional elements

An upper bound von Mises stress constraint is generated for the von Mises stress at the centroid of all the three-dimensional elements. The von Mises stress is constrained to be below the von Mises stress limit (SS) on the MAT1 material data entries. If a three-dimensional element references a MAT9 material, no automatic stress constraints will be generated. Note that if both three-dimensional elements and BAR elements from the

Shape and Sizing Design Capabilities

element library with shear stresses reference the same MAT1 data, erroneous constraints will be generated. This is because the three-dimensional elements require the von Mises shear stress limit while the BAR elements require the maximum shear stress limit. In this case, two different MAT1 entries must be used.

Meaning of ST, SC and SS in MAT1 data

	ST	SC	SS
ROD	Tension	Compression	- - -
GENERIC BAR	Tension	Compression	- - -
GENESIS LIBRARY BAR	Tension	Compression	Shear
USER LIBRARY BAR	- - -	- - -	- - -
BEAM	- - -	- - -	- - -
QUAD4 (PSHELL), TRIA3 (PSHELL), QUAD8 (PSHELL), TRIA6 (PSHELL)	- - -	- - -	von Mises
PCOMP/PCOMPG: HILL	X	Y	S
PCOMP/PCOMPG: HOFFMAN, TSAI-WU, STRN	XT, YT	XC, YC	S
SHEAR	- - -	- - -	Max shear
TRIAX6	- - -	- - -	von Mises
HEXA, HEX20, PENTA, PYRA TETRA	- - -	- - -	von Mises

3.3.9 Automatic Generation of Grid Stress Constraints

GENESIS has the capability to automatically generate grid stress constraints for all grids connected to TRIAX6, HEXA, HEX20, PENTA, PYRA and TETRA elements in every static LOADCASE and LOADCOM. This leads to a reduction in design input data preparation. When this option is requested, upper bound grid von Mises stress constraints are generated automatically using the shear stress limits (SS) from the MAT1 material data.

Automatic generation of grid stress constraints is requested using the DCONS input data. The format of this data used to request this feature is

1	2	3	4	5	6	7	8	9	10
DCONS	GSTRESS								

Shape and Sizing Design Capabilities

This command automatically generates DRESP1 data with the following characteristics:

1. The RID is always 99999.
2. The label is “AUTOMATC”.
3. The grid stress constraints are generated property by property. All the grids associated with each PSOLID and PAXIS property that references MAT1 data with shear stress limits have grid von Mises stress constraints.
4. If a grid is connected to two elements that have different properties, two constraints will be generated.
5. Each DRESP1 data is in a separate region.

DCONS data is generated for each DRESP1 with the same RID and “ALL” static LOADCASEs and LOADCOMs are flagged.

3.3.10 Automatic Generation of Surface Stress Constraints

Automatic grid stress constraints can be generated only for surface grids with the command:

1	2	3	4	5	6	7	8	9	10
DCONS	SURFACE								

This feature is the same as automatic generation of grid stress constraints, except that internal grids are ignored. Note that automatic generation of surface grid stress constraints only applies to grids connected to HEXA, HEX20, PENTA, PYRA and TETRA elements.

3.4 Advanced Shape Design Capabilities

This section describes the advanced shape design capabilities available in *GENESIS*.

3.4.1 Coordinate System Definition

When coordinate systems are defined by three grid point locations (CORD1R, CORD1C, and CORD1S data) the orientation of these coordinate systems does not change during the design process. This is true even if the grid points are involved in the shape design problem. In fact, the coordinate systems are defined using the grid point locations on the GRID data which may be different than the initial grid point locations. The initial grid point locations can be different than the locations specified on the GRID data if the initial values of the shape problem design variables are nonzero.

The exception to this rule is the definition of the element coordinate systems (for bar, plate/shell, and solid elements). In this case, the element coordinate systems are generated using the current grid point locations. Note that the BAR element coordinate system can be made to change with the shape of the structure by defining the orientation vector with a grid point that is involved in the shape problem. Note also that this grid point does not have to be connected to any structural elements.

3.4.2 Move Limits

There are four types of move limits that effect the shape design problem. These are shape design variable move limits, move limits on grid point locations during each design cycle, upper and lower bounds on grid point locations, and pass through constraints generated using DRESP2, DRESP3 or DRESPG data.

The design variables that control the shape problem have two characteristics that separate them from the sizing design variables. The first is that they usually have an initial value of zero. This is so that the initial design is the same as that specified on the GRID data. The second characteristic that differentiates the shape design variables from the sizing design variables is that they can take on negative values and may switch between positive and negative values during the design process.

Because the design variables may cross zero, the minimum move limit DXMIN becomes very important. If only the fractional move limit DELX was used, the design variables would not be able to cross zero. If the initial value of the design variable is zero then the default value of the minimum move limit is 0.1. The basis or grid perturbation vectors should be constructed so that a change of 0.1 in the design variable corresponds to about a 10% change in location of the grid points from their initial locations to their greatest possible optimum locations. Since it may be difficult to scale the basis or grid perturbation vectors, another approach is to change the default value of DXMIN on the DVAR data to account for this 10% move limit. If the design is very sensitive to small changes in the value of the shape design variable, then a smaller minimum move must be used. To give an example of the effect of DELX and DXMIN on the design process consider the moves taken by a design variable as it goes from 1.0 to -1.0 with DELX equal to 0.5 and DXMIN equal to 0.1. The design variable history would be 1.0, 0.5, 0.25, 0.125, 0.025, -0.075, -0.175, -0.275, -0.413, -0.619, -0.928, -1.0. Note that this takes 11 design cycles. If the minimum move limit was 0.3 then the design variable history would be 1.0, 0.5, 0.2, -0.1, -0.4, -0.7, -1.0. In this case only 6 design cycles are required.

In many design situations the basis or grid perturbation vectors correspond to a sizing type quantity such as a thickness or a radius. In this case it is a good idea to let the shape design variable take on the value of the sizing quantity. This can be done by creating a dependent design variable that is equal to the shape design variable minus its original value. The dependent design variable is then used on the DVGRID data. With this formulation the initial value of the dependent design variable is zero and the initial design is the same as that on the GRID data. If this type of formulation is used, then the move limits on the independent design variable will have some physical significance. In this case, large move limits should be used for the dependent design variable so that they never become critical.

The second type of move limits that effect the shape design problem are those placed on the grid point locations during each design cycle. These are used to prevent large changes in the shape of the structure in regions where the responses are very sensitive to these changes. An example could be the radius of a fillet where there is a stress concentration. These move limits are specified on the GRID data with the move limit parameter MV equal to 1. Note that lower bound move limits should be negative.

When the basis vector approach is used, many grid point locations are, in a sense, linked together. Because of this linking it is only necessary to place move limits on selected grid point locations, rather than all of the grids in a specific area.

The third type of move limits are used to prevent unrealistic designs generated by having one part of the structure occupying an area where another part of the structure is located. Suppose that a structural component must fit in a box generated by the $X=0$, $X=10$, $Y=0$, and $Y=20$ planes (see figure below). In order to keep the structure from occupying space on the other side of the $X=10$ plane grid point 6 should be constrained to have its X coordinate less than 10.

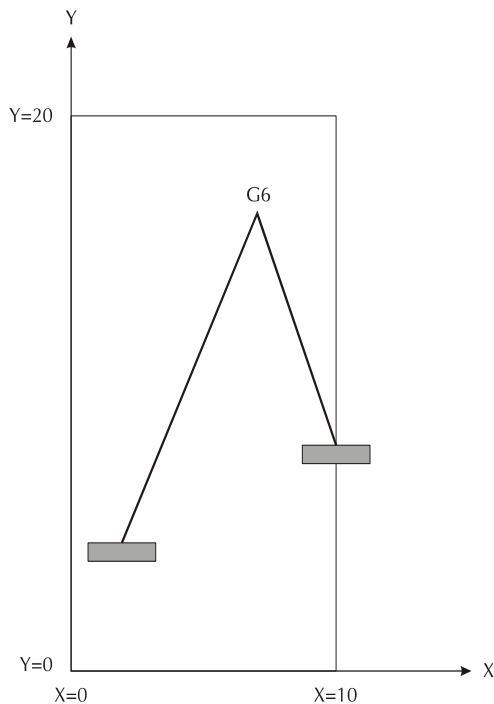


Figure 3-32Grid Move Limits

This is done using the move limits on the GRID data with the move limit parameter $MV=0$ (the default) as shown below.

	1	2	3	4	5	6	7	8	9	10
GRID	6	9.0	10.0							
+	0		10.0							

If the basis or grid perturbation vectors are generated such that other grid points could be located on the other side of the $X=10$ plane while grid 6 is not, then these grid points should also be constrained.

Shape and Sizing Design Capabilities

The last type of grid point move limit is used to prevent two parts of the structure from passing through each other during the design process (see **Figure 3-33**). This is done using the DRESP2 input data. The distance between the grid points on Face 1 and Face 2 of the structure should always be greater than 0.0. Assuming that the two faces will always be straight this can be achieved by constraining the distances (in the X direction) between grids 1 and 21 and between grids 19 and 39 to be greater than 0.

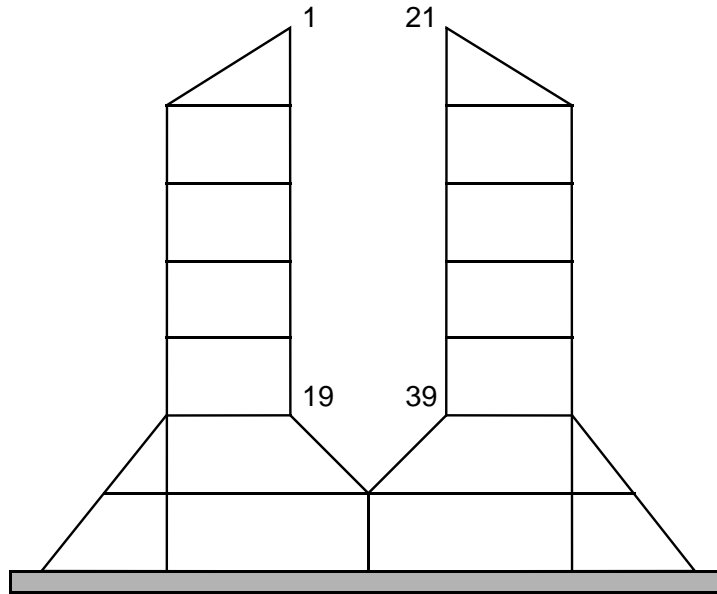


Figure 3-33Grid Pass Through

Note: Pass through constraints can easily be generated using DRESPG data.

The input data for these constraints would be:

	1	2	3	4	5	6	7	8	9	10
DRESP2	10	TOP	100							
+	DGRID	1	1	21	1					
DRESP2	20	BOTTOM	100							
+	DGRID	19	1	39	1					
DEQATN	100	$F(X1,X2) = X2-X1$								
DCONS	10	ALL	0.0	1.0E30						
DCONS	20	ALL	0.0	1.0E30						

Another example of a pass through constraint is given in [Equation Utility](#) (p. 271).

3.4.3 Effect of Nonrectangular Coordinate Systems

Care must be used when grid point locations are defined in nonrectangular coordinate systems in the shape problem for two reasons. The first is that when grid point perturbation vectors are used, the perturbations are not updated as the design changes. This means that if a perturbation is defined in an angular direction (θ or ϕ), the grid will move in a tangential direction rather than in a circumferential direction (see the figure below). The second reason that care must be taken involves the use of grid point move limits which are defined in the input (CP) coordinate system. If move limits are defined in the angular direction (θ or ϕ) and the grid point is located near the $\theta = 0$ or $\Phi = 0$ axis unexpected results may occur. This is because the angles less than 0 or greater than 360 degrees are not defined. Therefore, if a grid point location changes from $\theta = 1^\circ$ to $\theta = -1^\circ = 359^\circ$, it will be interpreted as a move of 358° , rather than 2° . If a grid point is located near an axis and requires an angular move limit, it is suggested that its input coordinate system be redefined in an alternate local coordinate system.

Care must also be taken when the grid point displacements are defined in a nonrectangular coordinate system. This is because unexpected results may occur if displacement constraints are used on grid points near the coordinate system origin. If the grid point location changes just slightly, the definition of the radial or angular direction may vary greatly. In fact, if the grid point is located at the origin of the coordinate system, then the angular directions are not defined. It is suggested the displacement constraints not be applied to grid points that are located near the origin of their output (CD) coordinate system.

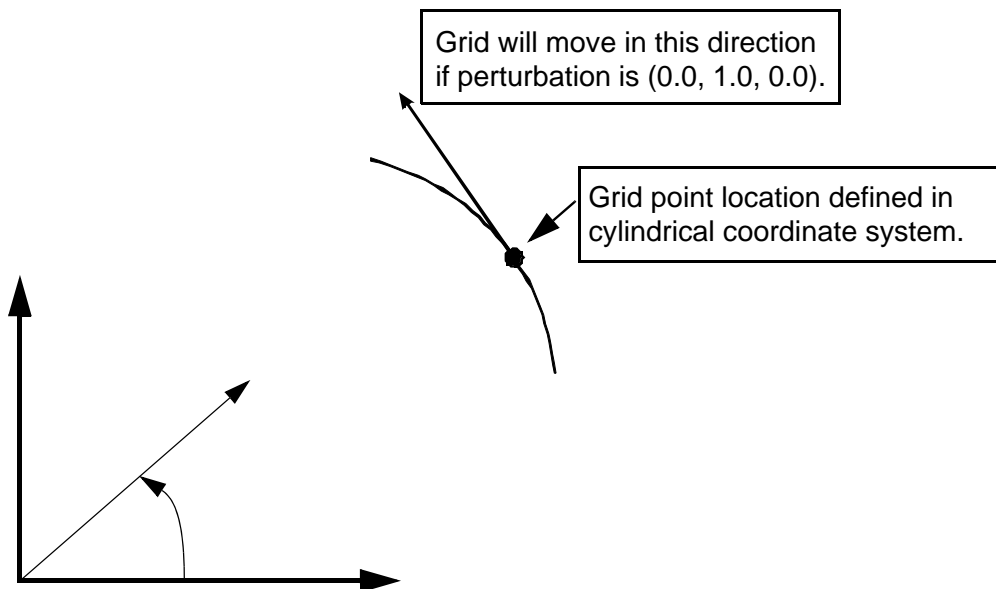
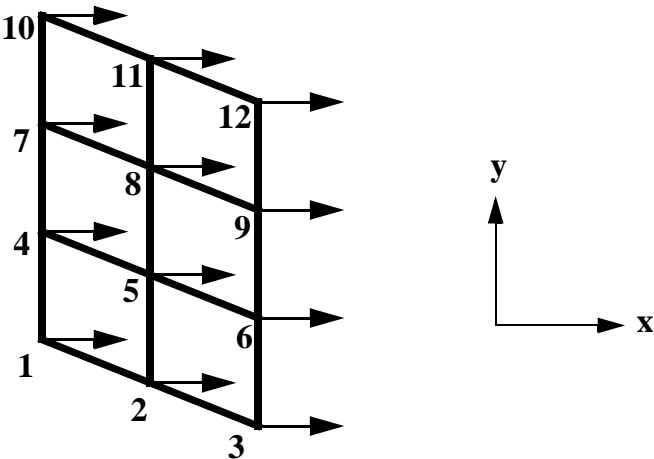


Figure 3-34 Non-Rectangular Coordinate System

3.4.4 Automatic Generation of Perturbation Vectors in *GENESIS*

Perturbation vectors are input in *GENESIS* using the DVGRID data statements. This data provides the perturbation of a single grid and the id of the design variable associated to it. For example, assume that a 12 node structure is going to be designed using one perturbation vector. The input data for the shape will include 12 GRIDS, 12 DVGRIDs and 1 DVAR data. The figure below shows the structure and the perturbation vector:



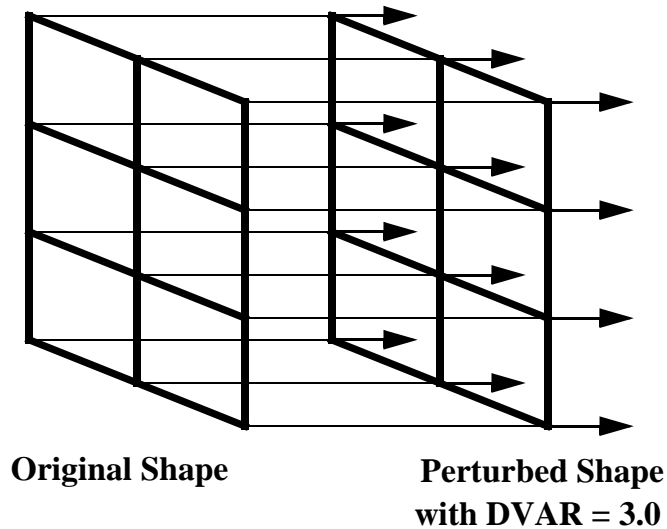
Perturbation Vector

The design input data could be:

	1	2	3	4	5	6	7	8	9	10
DVGRID	7	1				1.0	0.0	0.0	0	
DVGRID	7	2				1.0	0.0	0.0	0	
DVGRID	7	3				1.0	0.0	0.0	0	
DVGRID	7	4				1.0	0.0	0.0	0	
DVGRID	7	5				1.0	0.0	0.0	0	
DVGRID	7	6				1.0	0.0	0.0	0	
DVGRID	7	7				1.0	0.0	0.0	0	
DVGRID	7	8				1.0	0.0	0.0	0	
DVGRID	7	9				1.0	0.0	0.0	0	
DVGRID	7	10				1.0	0.0	0.0	0	
DVGRID	7	11				1.0	0.0	0.0	0	
DVGRID	7	12				1.0	0.0	0.0	0	
DVAR	7			0.0	-4.0	4.0				

Shape and Sizing Design Capabilities

During optimization the design variable could take different values giving different shapes to the structure. If the design variable value is zero then the current shape of the structure will be the same shape of the original design. If the design variable is one the shape of the current structure will be the shape of the a basis vector defined as the sum of the original design and the perturbation vector. If the design variable is for example, 3.0, the current shape will be as is shown in the figure below.



Creating DVGRID Data

The creation of the **DVGRID** data can be time consuming, so, *GENESIS* provides four ways to automatically generate perturbation vectors. The first method is based on interpolations (Geometric basis vector). The second method uses loads and boundary conditions (Natural basis vector). The third method converts grids into basis vectors. The fourth method uses the **DTGRID** data (topography optimization).

3.4.5 Geometric Perturbation Vectors (DOMAIN and DVGRIDC Data)

The procedure to use it is as follows: First, define a domain in which the design variable can act. A domain consists of 5 pieces of information; a DOMAIN ID, a TYPE to define the shape (and to mathematically define interpolation functions), the corners of the domain, a list of all non corner nodes of the domain (and the corners if user wants to repeat this information here), and a list of design variables.

This input data is called **DOMAIN** and its format is presented below:

1	2	3	4	5	6	7	8	9	10
DOMAIN	ID								
+	TYPE	CG1	CG2	CG3	CG4	CG5	CG6	CG7	CG8
+	"DGRID"	G1	G2	G3	G4	G5	G6	G7	G8
+		G9	G10				
+	"DVAR"	DV1	DV2	DV3	DV4	DV5	DV6	DV7	DV8
+		DV9	DV10				

Alternate Format:

1	2	3	4	5	6	7	8	9	10
DOMAIN	ID								
+	TYPE	CG1	CG2	CG3	CG4	CG5	CG6	CG7	CG8
+	"GSET"	SETID							
+	"DVAR"	DV1	DV2	DV3	DV4	DV5	DV6	DV7	DV8
+		DV9	DV10				

Field Information Description

2	ID	Unique Domain Identification number. (integer > 0)
2	TYPE	Domain element type: RBE2, BAR, TRIA3, QUAD4, BARX, TRIAX3, QUADX4, TETRA, PENTA, HEXA
3, 4, ...	CGI	Corner Grid Ids (Integer >0).
2	DGRID	Word indicating grid numbers will follow.
3, 4, ...	GI	Interior Grid Ids (Integer >0 or blank).
2	DSET	Word indicating grid set number will follow.
3	SETID	Set ID (Integer > 0)
2	DVAR	Word indicating design variable numbers will follow. See Remark 11.
3, 4, ...	DVI	DVAR entry identification number (Integer > 0)

The list of design variables is optional. If it is omitted, *GENESIS* will use all shape variables with this DOMAIN. The list of internal grid is also optional.

The second type of input data, required to automatically generate DVGRID data, is the DVGRIDC data statement. This data includes the design variable that controls this perturbation, the grid location of one grid and the perturbation.

In addition, as it will be explained below, this data includes information for generating perturbation vectors.

Shape and Sizing Design Capabilities

The basic format is:

1	2	3	4	5	6	7	8	9	10
DVGRIDC	DVID	GID	CID	COEFF	N1	N2	N3	BASIS	

Alternative Format:

1	2	3	4	5	6	7	8	9	10
DVGRIDC	DVID	GID	blank	COEFF	G1	G2	blank	BASIS	

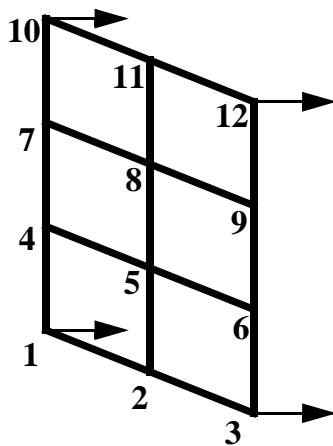
Field Information Description

2	DVID	DVAR identification number (Integer > 0).
3	GID	Grid number (Integer > 0). The grid should belong to a DOMAIN element. See Remark 1.
4	CID	Coordinate system identification number (Integer ≥ 0 or blank. Default = 0)
5	COEFF	Linear coefficients of relation between design variables and coordinates (Real, Default = 1.0. ignored if BASIS = 0).
6-8	N1,N2,N3	If BASIS = 0, components of a vector measured in the coordinate system defined by CID at the location of the grid defined by GID. If BASIS = 1, the location of the basis grid is in the CID coordinate system (Real or blank, Default = 0.0).
9	BASIS	BASIS/PERTUBATION Switch. BASIS Vector: BASIS = 1, Perturbation vectors: BASIS=0. (0, 1 or blank, Default is DOPT parameter BASIS).

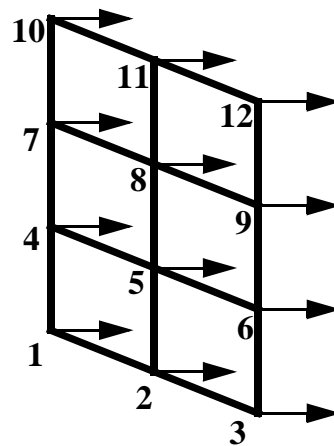
For example, to create a perturbation vector for the example shown in the previous figures, the following data could be used;

	1	2	3	4	5	6	7	8	9	10
DOMAIN	1									
+	QUAD4	1	3	12	10					
+	DGRID	1	2	3	4	5	6	7	8	
+		9	10	11	12					
+	DVAR	7								
DVGRIDC	7	1			1.0	0.0	0.0	0		
DVGRIDC	7	3			1.0	0.0	0.0	0		
DVGRIDC	7	12			1.0	0.0	0.0	0		
DVGRIDC	7	10			1.0	0.0	0.0	0		
DVAR	7		0.0	-4.0	4.0					

This data will internally (automatically) create the needed 12 DVGRIDS. Note that the word DVAR and the DVID 7 could be omitted.



Input Data



Perturbation Automatically Generated

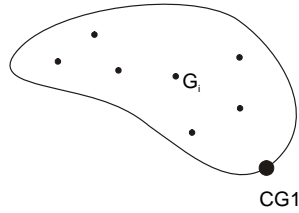
TYPES of DOMAIN Data and how Perturbations are Generated

There are 6 basic “elastic” domain shapes: linear (BAR), triangular (TRIA3), quadrilateral (QUAD4), tetrahedral (TETRA), pentahedral (PENTA) and hexahedral (HEXA). Also, there is one “rigid body” DOMAIN: RBE2 and three “axisymmetric” domain shapes: linear (BARX), triangular (TRIAX3) and quadrilateral (QUADX4).

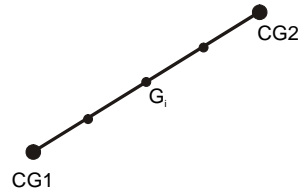
For the “elastic” and “axisymmetric” domains, the user specifies the basic perturbation on the corners of the DOMAIN or in a grid of the DOMAIN that is close to the middle of the corner grids. With that information *GENESIS* calculates the perturbation of the rest of grids in the DOMAIN using isoparametric interpolation functions. The interpolation inside a domain are made linear if only corner nodes are perturbed or are done quadratic in edges that have their middle node perturbed.

For the rigid body domain, the user specifies the perturbation in the corner node of the RBE2 domain. For convenience, it is allowed to add the perturbation to any node of the RBE2. If more than one perturbation is applied to the RBE2 domain, *GENESIS* will use the average perturbation for all the nodes in the domain and a warning message will be output.

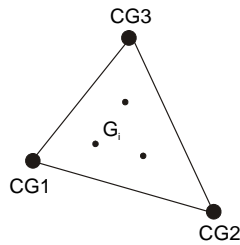
The following figures show the basic shapes of the DOMAIN elements.



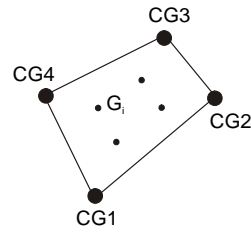
DOMAIN RBE2



DOMAIN BAR



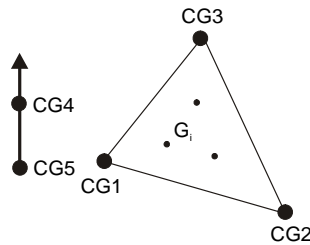
DOMAIN TRIA3



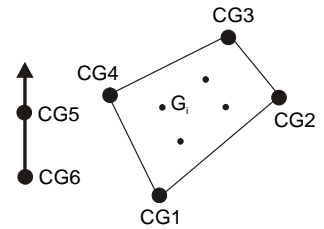
DOMAIN QUAD4



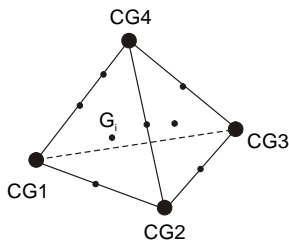
DOMAIN BARX



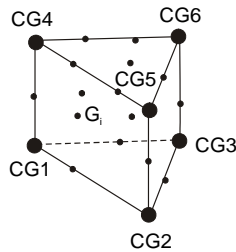
DOMAIN TRIAX3



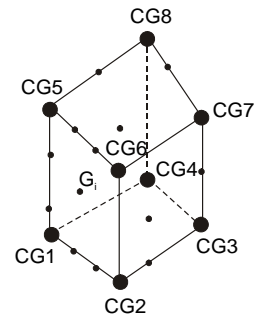
DOMAIN QUADX4



DOMAIN TETRA



DOMAIN PENTA



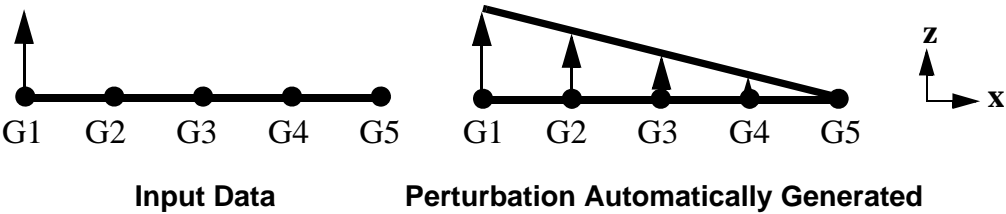
DOMAIN HEXA

Examples:

BAR DOMAIN

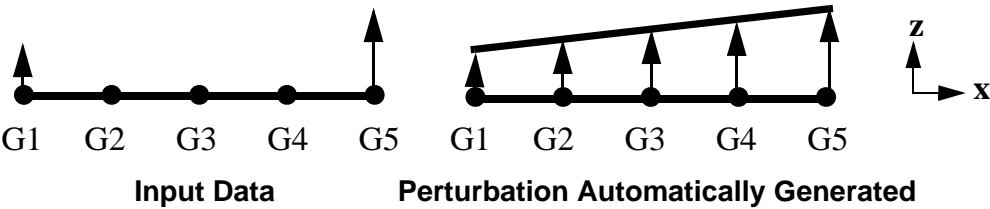
a) One corner perturbed

	1	2	3	4	5	6	7	8	9	10
DOMAIN	1									
+	BAR	1	5							
+	DGRID	1	2	3	4	5				
DVGRIDC	7	1			0.0	0.0	1.0	0		



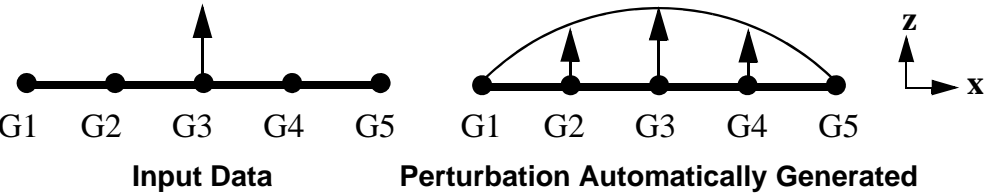
b) Two corners perturbed:

	1	2	3	4	5	6	7	8	9	10
DOMAIN	1									
+	BAR	1	5							
+	DGRID	1	2	3	4					
DVGRIDC	7	1			0.0	0.0	0.5	0		
DVGRIDC	7	5			0.0	0.0	1.0	0		



c) Middle node perturbed

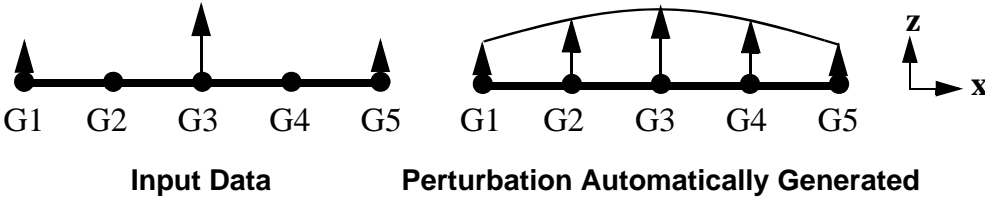
	1	2	3	4	5	6	7	8	9	10
DOMAIN	1									
+	BAR	1	5							
+	DGRID	1	2	3	4	5				
DVGRIDC	7	3			0.0	0.0	1.0	0		



3

d) All corners perturbed and middle node perturbed

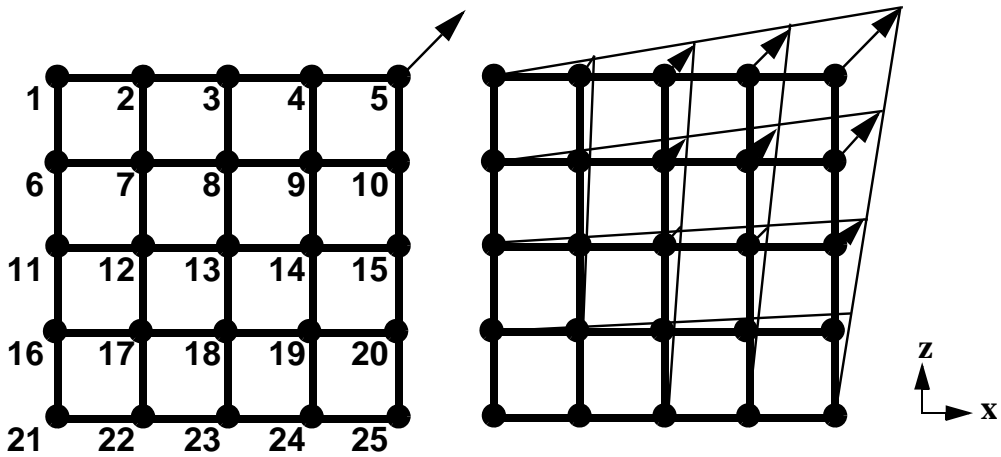
	1	2	3	4	5	6	7	8	9	10
DOMAIN	1									
+	BAR	1	5							
+	DGRID	1	2	3	4	5				
DVGRIDC	7	1			0.0	0.0	1.0	0		
DVGRIDC	7	3			0.0	0.0	2.0	0		
DVGRIDC	7	5			0.0	0.0	1.0	0		



QUAD4 DOMAIN

a) One corner perturbed;

	1	2	3	4	5	6	7	8	9	10
DOMAIN	1									
+	QUAD4	21	25	5	1					
+	DGRID	1	2	3	4	5	6	7	8	
+		9	10	11	12	13	14	15	16	
+		17	18	19	20	21	22	23	24	
+		25								
DVGRIDC	7	5				1.0	0.0	1.0	0	

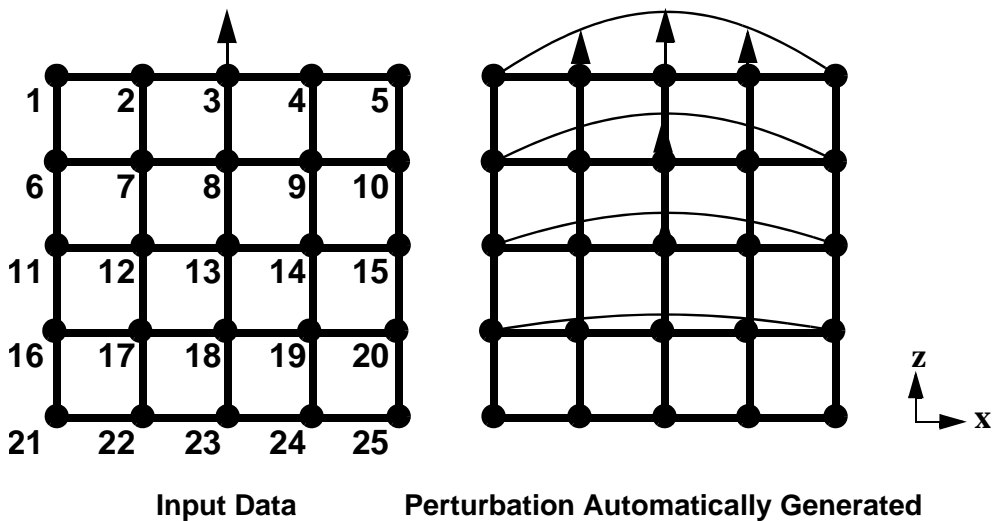


Input Data

Perturbation Automatically Generated

c) Midside node perturbed;

	1	2	3	4	5	6	7	8	9	10
DOMAIN	1									
+	QUAD4	21	25	5	1					
+	DGRID	1	2	3	4	5	6	7	8	
+		9	10	11	12	13	14	15	16	
+		17	18	19	20	21	22	23	24	
+		25								
DVGRIDC	7	3				0.0	0.0	1.0	0	

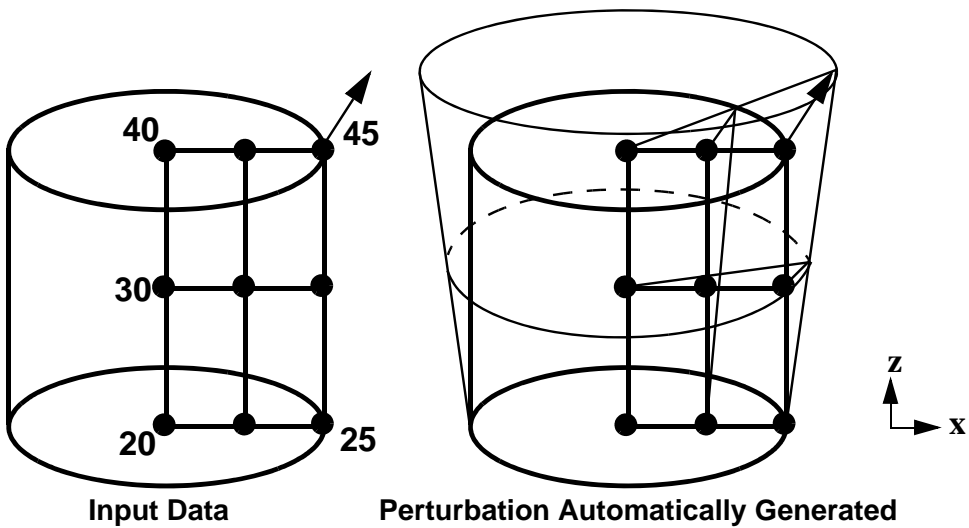


Note: Perturbations are not limited to inplane ones.

QUADX4 DOMAIN

a) One corner perturbed;

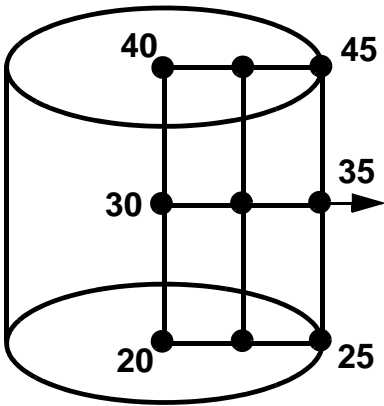
	1	2	3	4	5	6	7	8	9	10
DOMAIN	1									
+	QUADX4	20	25	45	40	30	20			
+	GSET	1								
DVGRIDC	7	45				1.0	0.0	1.0	0	



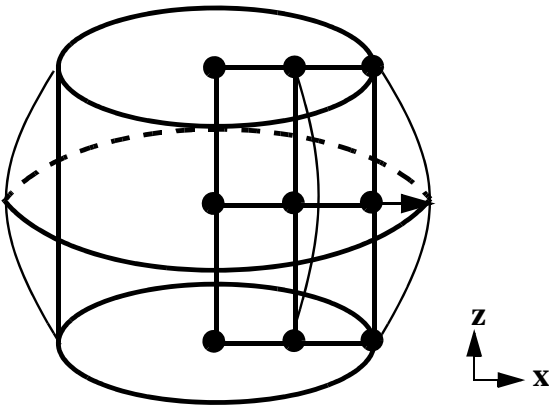
Shape and Sizing Design Capabilities

a) Midside node perturbed;

	1	2	3	4	5	6	7	8	9	10
DOMAIN	1									
+	QUADX4	20	25	45	40	30	20			
+	GSET	1								
DVGRIDC	7	35			1.0	0.0	0.0	0		



Input Data

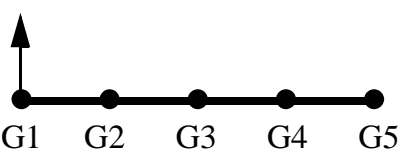


Perturbation Automatically Generated

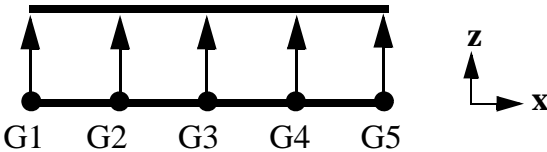
RBE2 DOMAIN

a) Corner perturbed;

	1	2	3	4	5	6	7	8	9	10
DOMAIN	1									
+	RBE2	1								
+	DGRID	1	2	3	4	5				
DVGRIDC	7	1			0.0	0.0	1.0	0		



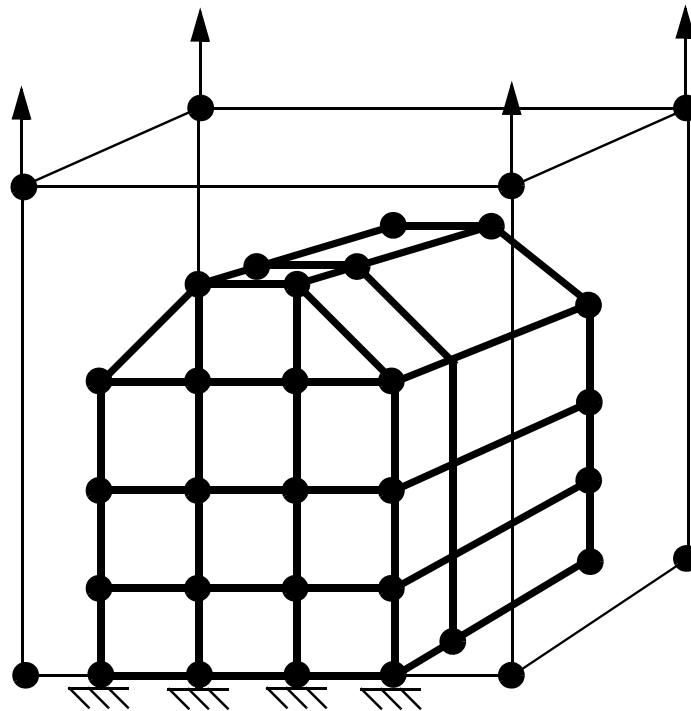
Input Data



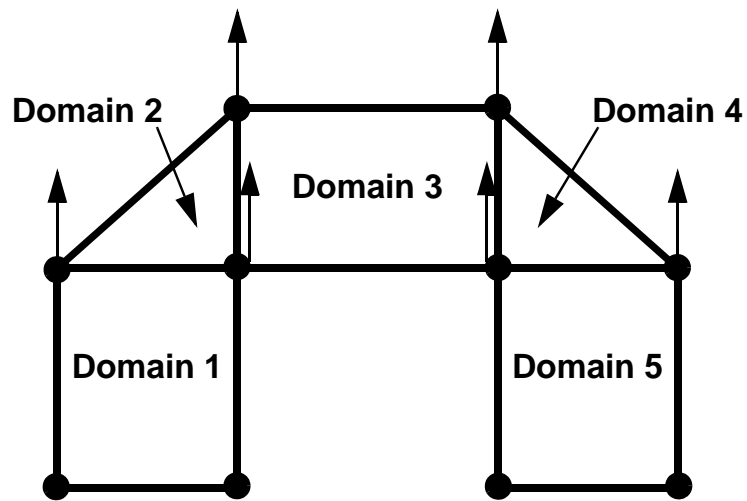
Perturbation Automatically Generated

Use of DOMAIN data:

The DOMAIN data can be used to create perturbation vectors for a part of the structure or for the whole. One or more domain elements can be used. A domain usually covers a group of elements, but that is not a requirement. In one extreme one DOMAIN element can be defined per element or in the other extreme one DOMAIN element can cover the whole structure. The DOMAIN element can also be “larger” than the structure. For example, one HEXA domain could be used to cover a complex structure.



One Domain (HEXA)



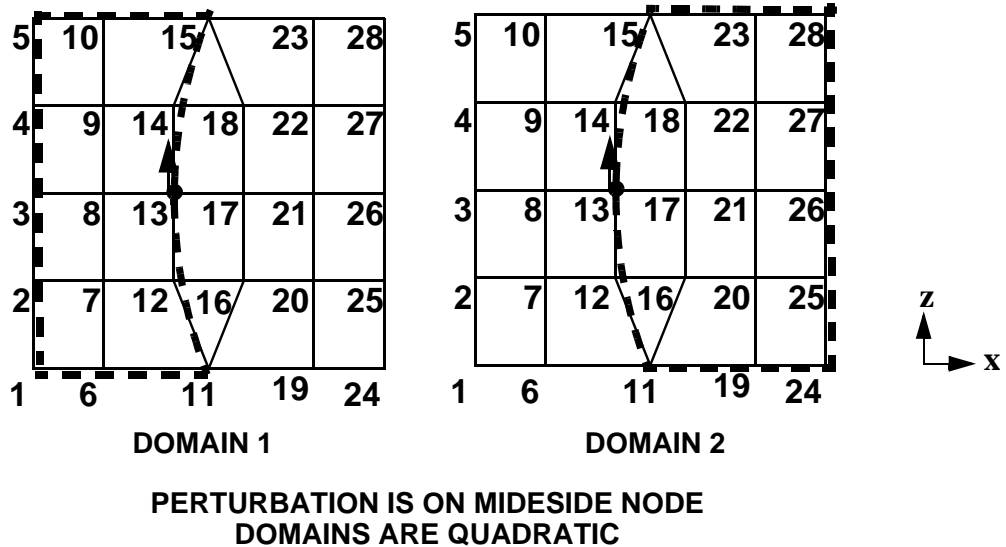
Multiple Domains

Treatment of Perturbation at Middle of the Edge of a DOMAIN

If a DVGRIDC data statement references a grid that is close to the middle of an edge of a DOMAIN, *GENESIS* will treat that grid as a grid of the DOMAIN element and use it as the location for a midside node. That midside node will define internally a quadratic interpolation function for that edge.

Example

In the following example, a perturbation is applied to grid 13, which belongs to DOMAIN 1 but not DOMAIN 2. However, because grid 13 is at the midside of an edge (defined by corner grids 11 and 15) that is shared by both domains, both domains are affected by the perturbation. *GENESIS* will make both domain elements have a quadratic edge, insuring that the grids on the edge are perturbed in a compatible way.



	1	2	3	4	5	6	7	8	9	10
DOMAIN	1									
+	QUAD4	1	11	15	5					
+	DGRID	1	2	3	4	5	6	7	8	
+		9	10	11	12	13	14	15		
DOMAIN	2									
+	QUAD4	11	24	28	15					
+	DGRID	11	16	17	18	15	19	20	21	
+		22	23	24	25	26	27	28		

3

If instead of a perturbation at grid 13, the perturbation is at grid 15, then the DOMAINS will be linear as shown in the figure below.



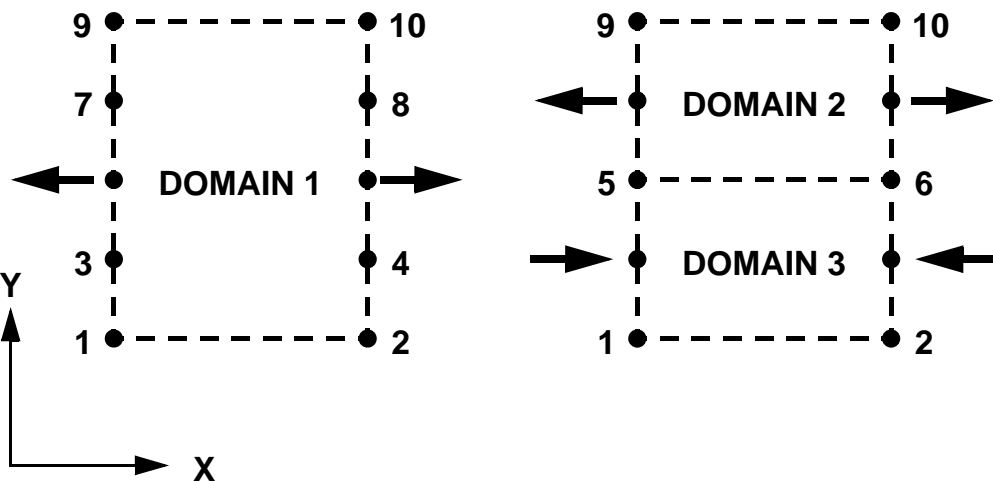
A diagram of a curved rectangular element. The vertical dimension is labeled L and the horizontal dimension is labeled A . The element is defined by a curved left boundary and straight right, top, and bottom boundaries. A grid of lines is shown within the element, with curved lines parallel to the left boundary and straight lines parallel to the other three boundaries. Arrows indicate the directions of the dimensions L and A .

A/L < DVG TOL

Using DVAR to Restrict the DOMAIN to a Specified Set of Design Variables

This allows you to overlap domains or to limit the scope of the design variables associated with a domain.

Example of Overlapping DOMAINS



3

Here three domains are used for one part of the structure.

DOMAIN 1 Data

	1	2	3	4	5	6	7	8	9	10
DOMAIN	1									
+	QUAD4	1	2	10	9					
+	DGRID	3	4	5	6	7	8			
+	DVAR	10								
DVGRIDC	10	5			-1.0	0.0	0.0	0		
DVGRIDC	10	6			1.0	0.0	0.0	0		

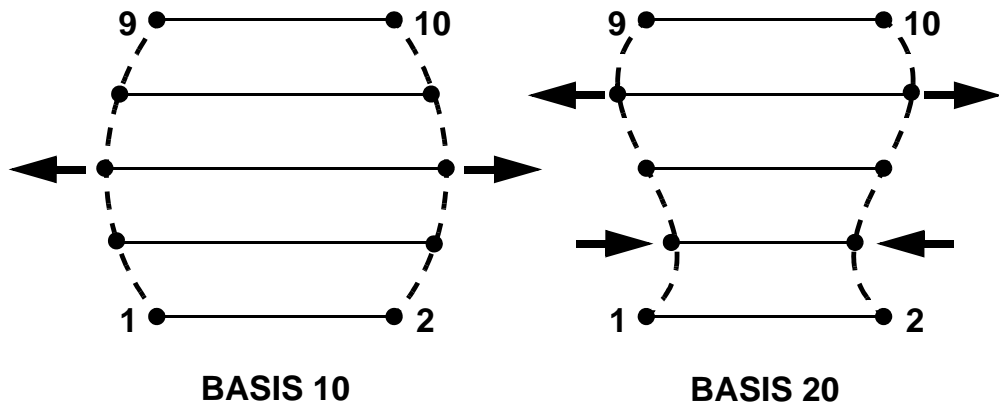
DOMAIN 2 Data

	1	2	3	4	5	6	7	8	9	10
DOMAIN	2									
+	QUAD4	5	6	10	9					
+	DGRID	7	8							
+	DVAR	20								
DVGRIDC	20	7				-1.0	0.0	0.0	0	
DVGRIDC	20	8				1.0	0.0	0.0	0	

DOMAIN 3 Data

	1	2	3	4	5	6	7	8	9	10
DOMAIN	3									
+	QUAD4	1	2	6	5					
+	DGRID	3	4							
+	DVAR	20								
DVGRIDC	20	3				1.0	0.0	0.0	0	
DVGRIDC	20	4				-1.0	0.0	0.0	0	

This data creates the following basis vectors.

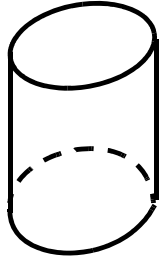


NOTE: If DOMAIN 1 is not restricted to use only DVAR 10, an error message will be generated because the DVGRIDC corresponding to DVAR 20 will have two perturbations per edge at the two vertical edges of DOMAIN 1. This is not allowed.

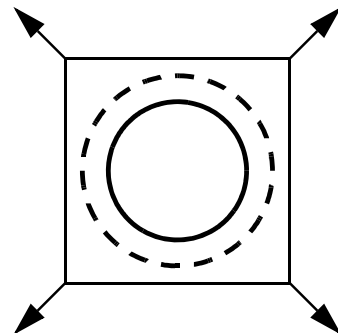
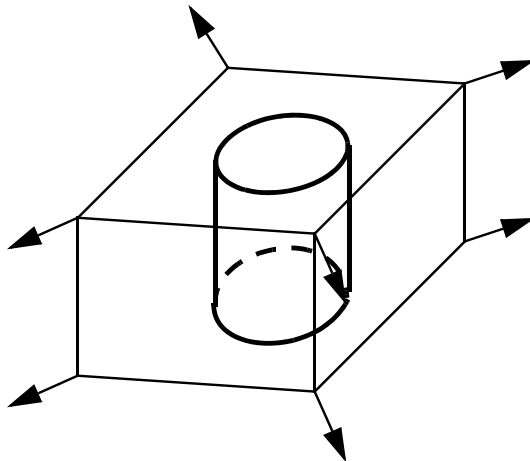
Since the numbers used in DVGRIDC are perturbations, the DOPT parameter BASIS used in this example should be BASIS = 0.

A Modeling Trick:

The DOMAIN does not have to match the FEM model. For example, assume we wish to design the diameter below.

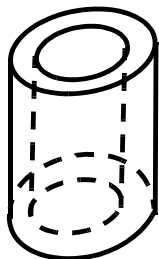


The DOMAIN can be a HEXA



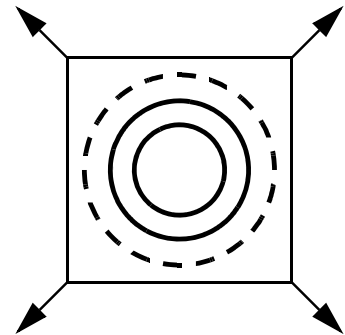
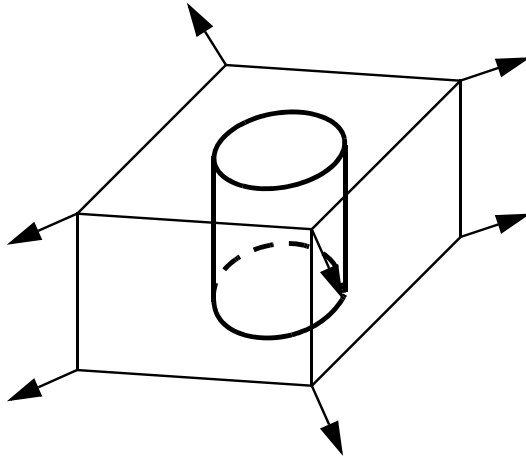
**BASIS VECTOR
TOP VIEW**

Now assume we wish to design the thickness of a cylinder wall.



Shape and Sizing Design Capabilities

The DOMAIN is the same as before. However, we do not include the grids on the inner wall of the cylinder. Then only the outer diameter will change.



**BASIS VECTOR
TOP VIEW**

Perturbation in Arbitrary Coordinate Systems and Scale Factors

The perturbation defined in DVGRIDC can be input in any coordinate system. Also, the perturbation can be scaled using a coefficient. This additional data is defined in the general format:

1	2	3	4	5	6	7	8	9	10
DVGRIDC	DVID	GID	CID	COEFF	N1	N2	N3	BASIS	

Field Information Description

2	DVID	DVAR identification number (Integer > 0).
3	GID	Grid number (Integer > 0). The grid should belong to a DOMAIN element. See Remark 1.
4	CID	Coordinate system identification number (Integer ≥ 0 or blank. Default = 0)
5	COEFF	Linear coefficients of relation between design variables and coordinates (Real, Default = 1.0. ignored if BASIS = 0).
6-8	N1,N2,N3	If BASIS = 0, components of a vector measured in the coordinate system defined by CID at the location of the grid defined by GID. If BASIS = 1, the location of the basis grid is in the CID coordinate system (Real or blank, Default = 0.0).
9	BASIS	BASIS/PERTUBATION Switch. BASIS Vector: BASIS = 1, Perturbation vectors: BASIS=0. (0, 1 or blank, Default is DOPT parameter BASIS).

Perturbation using two grid locations to define direction

The perturbation defined in DVGRIDC can be input by giving two additional grid ids; that provides a direction of the perturbation, the magnitude of the perturbation will be provided by coeff.

1	2	3	4	5	6	7	8	9	10
DVGRIDC	DVID	GID	blank	COEFF	G1	G2	blank		

Field Information Description

2	DVID	DVAR identification number (Integer > 0).
3	GID	Grid number (Integer > 0). The grid should belong to a DOMAIN element. See Remark 1.
4	CID	Coordinate system identification number (Integer ≥ 0 or blank. Default = 0)
5	COEFF	Linear coefficients of relation between design variables and coordinates (Real, Default = 1.0).
6	G1	Initial grid to define direction of perturbation
7	G2	Final grid to define direction of perturbation

Note: Even if the grids used to define the perturbation change during the optimization the perturbation will not. The perturbation is calculated only once at the beginning of the run.

Printing Perturbation Vectors

The Perturbation vectors created using DOMAIN and DVGRIDC data can be output using the Solution Control parameter DVGRID = PRINT. All Perturbation vectors can be output to the post-processing files using the Solution Control command PERTURBATION = POST.

3.4.6 Geometric Basis Vector (DOMAIN and DVGRIDC Data)

This procedure is used when field 9 is 1 (or field 9 is blank and DOPT parameter BASIS is 1).

The procedure to automatically create basis vectors is completely analogous to the process of creating perturbation vectors. However, only one major difference exists; The DVGRIDC data is used to create basis grid locations instead of perturbations.

The input data of DVGRIDC in this case is:

1	2	3	4	5	6	7	8	9	10
DVGRIDC	DVID	GID	CID	blank	N1	N2	N3	1	

Field Information Description

2	DVID	DVAR identification number (Integer > 0).
3	GID	Grid number (Integer > 0). The grid should belong to a DOMAIN element. See Remark 1.
4	CID	Coordinate system identification number (Integer ≥ 0 or blank. Default = 0)
6-8	N1,N2,N3	Location of the basis grid is in the CID coordinate system (Real or blank, Default = 0.0).

GENESIS will subtract from N1, N2 and N3 the coordinates of grid GID to create a corresponding perturbation. After that, it will use the perturbations to calculate the perturbation of all grids in the DOMAINS and then add to these perturbations the grid coordinates to construct DVGRIDs with basis grid locations.

Printing Basis Vectors

The Basis vectors created using DOMAIN and DVGRIDC data can be output using the Solution Control parameter DVGRID = PRINT. All Basis vectors can be output to the post-processing files using the Solution Control command BASIS = POST.

3.4.7 Natural Perturbation Vectors (DVSHAPE Data)

The natural perturbation vector method provides a way to simplify the creation of perturbation vectors for shape optimization by using displacement fields as perturbation fields.

The procedure to optimize a structure using natural perturbation vectors is as follows:

1. Create an auxiliary model that contains loads and boundary conditions that produce displacement patterns to be used as perturbations.
2. Add DVSHAPE entries to the auxiliary model for every desired perturbation vector.
3. Run *GENESIS* using the auxiliary model.
4. Add the “*.DVS” file created by *GENESIS* to the original model.
5. Run the original model.

Note: The DOPT parameter BASIS has to be set to 0 in both the auxiliary file and the in the original file.

The Auxiliary Model

The auxiliary model is typically a modified copy of the original model, and its purpose is to create a set of displacements to be used as perturbation vectors. The auxiliary model must contain the following:

1. Loads and/or boundary conditions to produce desired displacements. Typically SPCD are used, though any type of static loads are acceptable. Temperature loads may be useful.
2. One or more DVSHAPE entries, each of which identifies:
 - a. A design variable and its bounds.
 - b. A set of grids for which DVGRID data is to be generated. If all grids are desired then the “ALL” option is allowed.
 - c. A maximum perturbation size. If this data is omitted, the perturbations will equal the translational displacements, otherwise, the displacements will be scaled accordingly.
3. DOPT parameter BASIS set to 0 or 1. For perturbation, a value of 0 is needed.

If desired, the auxiliary model can also contain:

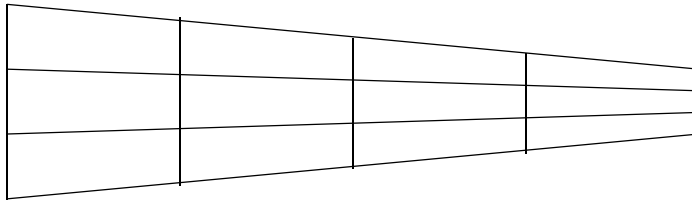
4. Boundary conditions to limit the scope of the perturbation.
5. Additional elements to produce desired displacements.
6. Special properties and/or materials to produce special conditions.
For example, low Poisson’s ratio to avoid longitudinal and transverse coupling.

Example:

Consider the following structure:

Shape and Sizing Design Capabilities

Assume it is desired to create a perturbation vector for the plate structure in figure.



There are several ways to do it. One is to enforce displacements that match the desired perturbations along the boundary.

To do that the following loadcase has to be created:

```
LOADCASE 10
  LOAD = 100
  SPC = 200
  DVSHAPE = 10,-1.0,2.0,ALL
```

Where,

LOAD = 100 Select SPCD bulk data that produce the desired displacements on the boundary

SPC = 200 Select SPC1 bulk data that produce the desired boundary conditions

DVSHAPE = 10,-1.0,2.0,ALL Creates a design variable (id=10) with initial value of zero, lower bound of -1.0, and upper bound of 2.0. ALL indicates that all grids with nonzero displacement are selected for the perturbation vector.

The bulk data has to contain the necessary SPCD and SPC1 data.

Multiple DVSHAPE entries are allowed per loadcase. In that way multiple perturbation vectors can be created for each loadcase in the auxiliary model.

To obtain the perturbation vectors, simply run *GENESIS*. The generated DVGRID entries, as well as the associated DVAR entries, are stored in a file called “pname_aux.DVS” where pname_aux is the project name of the auxiliary file.

To print the perturbation vectors in the output file, the solution control DVGRID=PRINT can be used.

The Original Model

To use the results on the original model, the file `pname_aux.DVS` has to be included in the bulk data of the original model. This is done by adding an `INCLUDE` entry as follows:

```
INCLUDE 'pname_aux.DVS'
```

Alternatively, the contents of the “*.DVS” file can be directly copied into the file of the original model using any text editor.

To print the perturbation vectors to a post processing file the command `PERTURBATION = POST` can be used. The name of the post processing file is `pname.DVG`, where `pname` is the project name of the original model.

The original model can also contain `DOMAIN`, `DVGRIDC` and `DVGRID` data. In other words, natural perturbation vectors can be mixed with geometric and manual perturbation vectors.

Multiple auxiliary files can be used. In this case, multiple `INCLUDE` entries would be used.

3.4.8 Natural Basis Vectors (DVSHAPE Data)

The procedure to create and use natural basis vectors is the same as the procedure to create and use natural perturbation vectors. The only difference is that the DOPT parameter BASIS has to be set to 1 in both the auxiliary file and the original file.

3.4.9 Grid Basis Vectors (DVBASIS Data)

The grid basis vector method provides a way to create basis vectors using a preprocessor that can perturb grids. The grids then are transformed into basis vectors.

The procedure to optimize a structure using GRID BASIS vector is described next:

1. Create an auxiliary model that has perturbed grids.
2. Add DVBASIS entries to the auxiliary model for every desired basis vector.
3. Run *GENESIS* using the auxiliary model (CHECK mode is sufficient).
4. Add the “*.DVB” file created by *GENESIS* to the original model.
5. Run the original model.

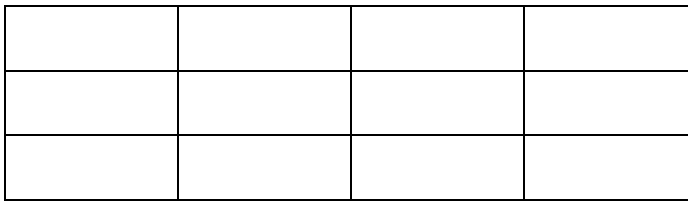
The Auxiliary Model

The auxiliary model is typically a modified copy of the original model, and its purpose is to create sets of grid that are modifications of the original grids. The auxiliary model must contain the following:

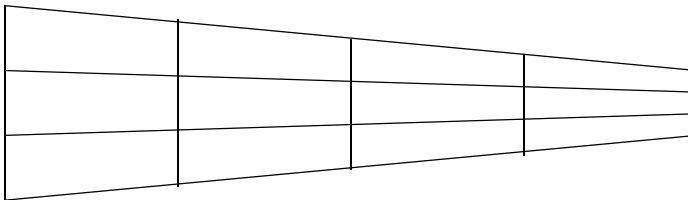
1. A valid input data.
2. One or more DVBASIS entries, each of which identifies:
 - a. A design variable and its bounds.
 - b. A set of grids for which DVGRID data is to be generated. If all grids are desired then the “ALL” option is allowed.

Example:

Consider the following structure:



Assume it is desired to create a basis vector for the plate structure in figure.



There are several ways to do it. One is to perturb the grid using a preprocessor that allows to stretch the model.

After the model is finished, add the following data to the solution control:

DVBASIS = 10,-1.0,2.0,ALL

This data will:

creates a design variable (id=10)
with initial value of zero,
lower bound of -1.0, and upper bound
of 2.0. ALL indicates that all grids
with nonzero displacement are selected
for the perturbation vector.

Multiple DVBASIS entries are allowed per auxiliary file. In that way multiple basis vectors can be created for each loadcase in the auxiliary model.

To obtain the basis vectors, simply run *GENESIS*. The generated DVGRID entries, as well as the associated DVAR entries, are stored in a file called “*pname_aux.DVB*” where *pname_aux* is the project name of the auxiliary file.

To print the perturbation vectors in the output file, the solution control DVGRID=PRINT can be used.

The Original Model

To use the results on the original model, the file *pname_aux.DVB* has to be included in the bulk data of the original model. This is done by adding an INCLUDE entry as follows:

```
INCLUDE 'pname_aux.DVB'
```

Alternatively, the contents of the “*.DVB” file can be directly copied into the file of the original model using any text editor.

To print the basis vectors to a post processing file the command BASIS = POST can be used. The name of the post processing file is *pname.DVG*, where *pname* is the project name of the original model.

The original model can also contain DOMAIN, DVGRIDC and DVGRID data. In other words, GRID basis vectors can be mixed with geometric and manual basis/perturbation vectors.

Multiple auxiliary files can be used. In this case, multiple INCLUDE entries would be used.

3.5 Reliability Based Optimization

Reliability Analysis

To perform reliability analysis, design variables must be assigned a random distribution. The assignment of a random distribution creates a "random design variable" and is done with the VTYPE and VVALUE fields on the **DVAR** data entry.

Details on this can be found in **Random Variables for Reliability Analysis** (p. 267).

Reliability Based Optimization

To perform reliability optimization, one or more constraints must be assigned an allowable probability of failure. Allowable probability of failure is specified by the PFi fields on the **DCONS** and/or **DCONS2** data entries.

More details on this can be found in **Allowable Probability of Failure on Constraint Entries** (p. 329).

Reliability Process

In a typical reliability optimization problem, *GENESIS* will first perform deterministic (traditional) optimization and after converging to a deterministic answer, it will automatically switch to probabilistic optimization. In the deterministic mode, the allowable probability of failures for constraints are not used. There are alternative starting points for the reliability process that can be selected by using the DOPT parameter **RBOST**.

Reliability Results

The solution control command **RELIABILITY** controls the amount of reliability results printed to the output file.

Whenever reliability optimization is performed, reliability results associated to element results will be printed in a file named *pnameRELxx.pch*. Where, *pname* is the *GENESIS* project name and *xx* corresponds to the corresponding design cycle number. This is a post-processing file using the PUNCH element strain energy format, where the 3 values per element are assigned alternative meanings. The first value is the maximum probability of failure of any retained constraint associated to that element in any loadcase. The second value is the maximum probability of failure index of any retained constraint associated to that element in any loadcase. The third value is the loadcase ID of the loadcase where the maximum probability of failure index of any retained constraint associated to that element occurs or 0 if none. Note that the probability of failure index is calculated as the probability of failure divided by the allowable probability of failure. A value greater than 1.0 for the probability of failure index means that the element exceeds the allowable.

3.6 Freeform Optimization

Introduction to Freeform Optimization

Freeform optimization is a special type of shape optimization. In its simplest form splits a user provided perturbation vector into multiple perturbation vectors. The intention is to increase the design freedom with little effort from part of the user. Freeform perturbation vectors can be used with any other shape, topography and/or sizing design data.

Freeform optimization is controlled with the **DSHAPE** data entry.

3.6.1 DSHAPE Data Entry

The **DSHAPE** entry provides ways to define attributes of a perturbation vector and/or to define a freeform shape optimization region.

The information specified by a DSHAPE entry is as follows:

1. Select a perturbation vector (**DVGRID** or **DVGRIDC**) by selecting its associated design variable. The grids associated to the perturbation correspond to the designable region.
2. Select if the perturbation vector should be freeform (FREE) or not (LINKED)
3. Select if only the control perturbations are split (FTYPE=DVGRIDC) or if all input and generated perturbation are split (FTYPE=DVGRID).
4. Optionally, define if the perturbation vector is scaled to set a maximum perturbation.
5. Optionally, define if the initial value of the design variable should be randomized.
6. Optionally, defined if the sensitivities associated to the perturbations vectors should be scaled.
7. If the perturbation vector is freeform, optionally create manufacturing constraints.
8. If the perturbation vector is freeform, optionally set a coarse diameter.
9. If the perturbation vector is freeform, optionally control variability.

3.6.2 Changing Attributes of a Perturbation Vector

The DSHAPE data can be used to change certain attributes of a shape perturbation set.

The basic format for DSHAPE for changing attributes is:

1	2	3	4	5	6	7	8	9	10
DSHAPE	DVARID	LABEL			MAXPERT	RINIT	SCALE		

In the above format, the DVARID is a unique identification number of the design variable associated to DVGRID and/or DVGRIDC entries. DVARID must not be left blank. The LABEL is optional and can be used to identify the perturbation vector. MAXPERT controls the maximum distance a grid will move when the design variable is at its lower or upper bound. RINIT sets the range for randomness in the design variable initial value. SCALE scales the perturbation and the design variable bounds to increase or decrease the sensitivity of the responses to the shape change.

Example 1:

The following data will ensure that all grids associated to perturbation 10 will not move more than 5.0 units (due to this perturbation set - other perturbation vectors may also move these grids). :

1	2	3	4	5	6	7	8	9	10
DSHAPE	DVARID	LABEL			MAXPERT				
DSHAPE	10	Top			5.0				

Example 2:

The following data will change the initial value of design variable 10 to be a random number. :

1	2	3	4	5	6	7	8	9	10
DSHAPE	DVARID	LABEL				RINIT			
DSHAPE	10	Top				1.0			

Note: RINIT is not needed for perturbations that cause change of volume. RINIT is typically needed for designing beads on flat plates.

Example 3:

The following data will multiply the perturbation by SCALE and divide the bounds of design variable 30 by SCALE:

1	2	3	4	5	6	7	8	9	10
DSHAPE	DVARID	LABEL					SCALE		
DSHAPE	10	Top					2.0		

Note: This effectively scales the sensitivities. This accelerates or slows down the movement of the design variables in a given design cycle. If SCALE is greater than 1.0 then the process is accelerated. This is used when the optimization seems to be too slow. A SCALE value lower than 1.0 allows the process to be slowed down to prevent mesh distortion or premature convergence.

3.6.3 Freeform with DVGRID data

The DSHAPE data can be used to split DVGRID data. The basic format for DSHAPE in this case is:

1	2	3	4	5	6	7	8	9	10
DSHAPE	DVARID	LABEL	SPLIT	FTYPE					

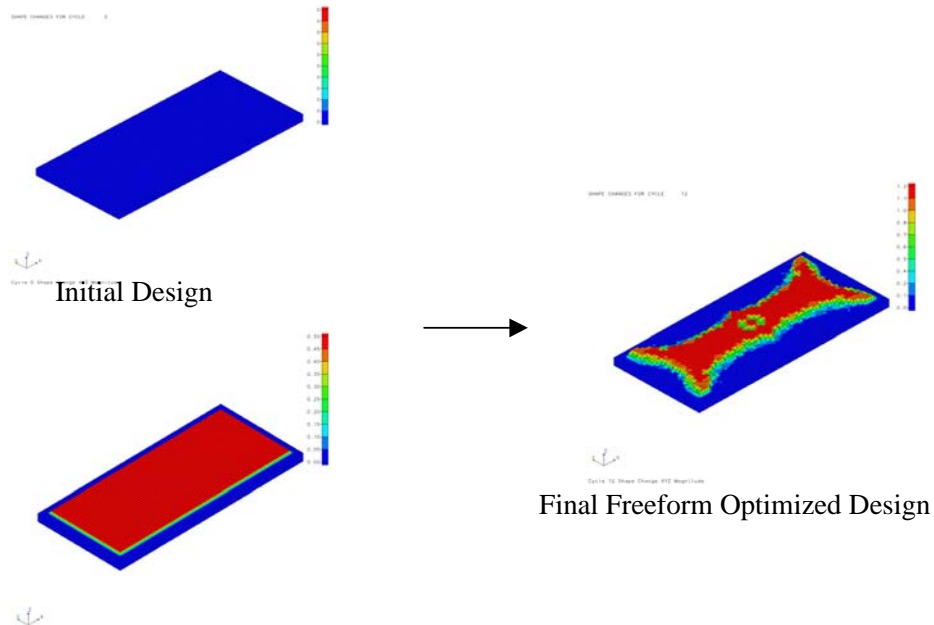
In the above format, the DVARID is a unique identification number of the design variable. The LABEL is used to help the user identify the perturbation vector. SPLIT is used to define whether we want to use freeform. A value of FREE means that we want to freeform. If SPLIT is blank or LINKED then no freeform will be performed with this DVARID. If SPLIT is FREE and FTYPE is DVGRID or blank, then GENESIS will split all input and generated perturbations such that they point to new independent copies of the design variable.

Example 4: Freeform with DVGRID data

The following data will change the initial value of design variable 10 to be a random number. : .

1	2	3	4	5	6	7	8	9	10
DSHAPE	DVARID	LABEL	SPLIT	FTYPE					
DSHAPE	40	Side	FREE	DVGRID					

The following example shows a structure that is modeled with solid elements. Originally the structure is flat. In this example, freeform finds the optimal places to add solid ribs.



3

Only One User Supplied Perturbation Vector Required

The values RINIT, MAXPERT and SCALE can also be use with FTYPE=DVGRID.

When FTYPE=DVGRID the program split the perturbation vector. Each of the new perturbation vectors gets its own new independent copy of the design variable. If fabrication constraints are present, the perturbation vectors new design variables will be linked according to the constraints. Using RINIT causes each of the internally created design variables to get its own distinct random initial value.

If the DVARID references a control perturbation (DVGRIDC) the program will split perturbations generated by the DOMAIN elements. In other words, the program processes the DVGRIDC normally to generate DVGRID and then freeform split the automatically created DVGRID data.

3.6.4 Freeform with DVGRIDC data

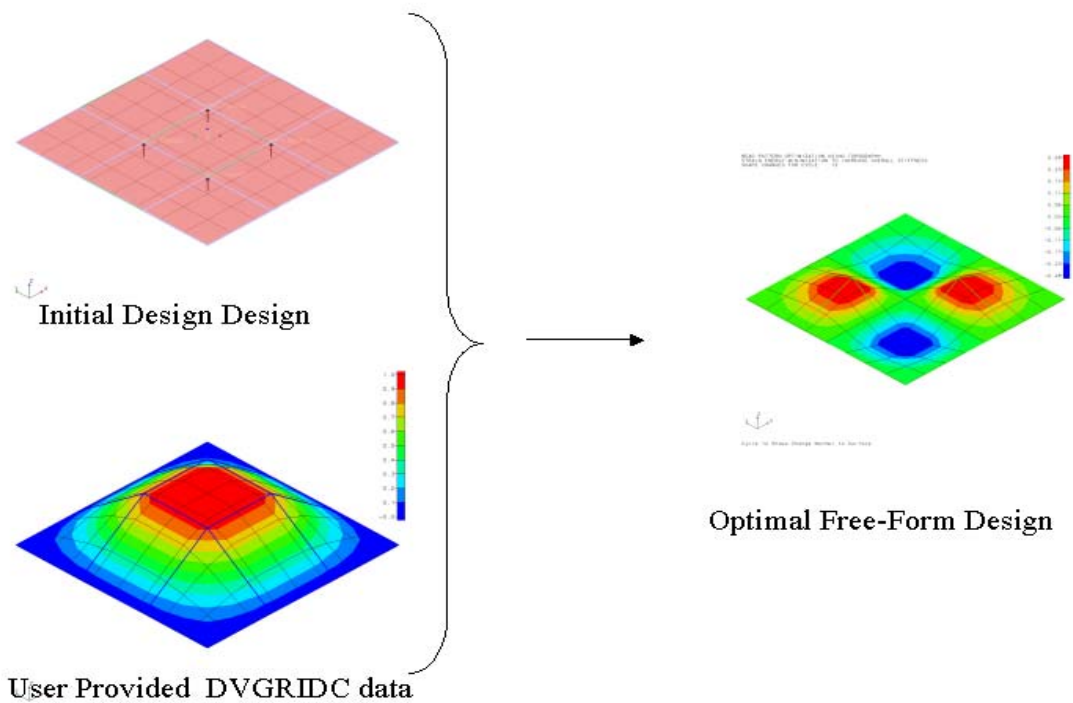
If SPLIT is FREE and FTYPE is DVGRIDC in the basic format described in the previous section, then GENESIS will split the control perturbations (DVGRIDC) into multiple control perturbations (DVGRIDC).

Example 5: Freeform with DVGRIDC data

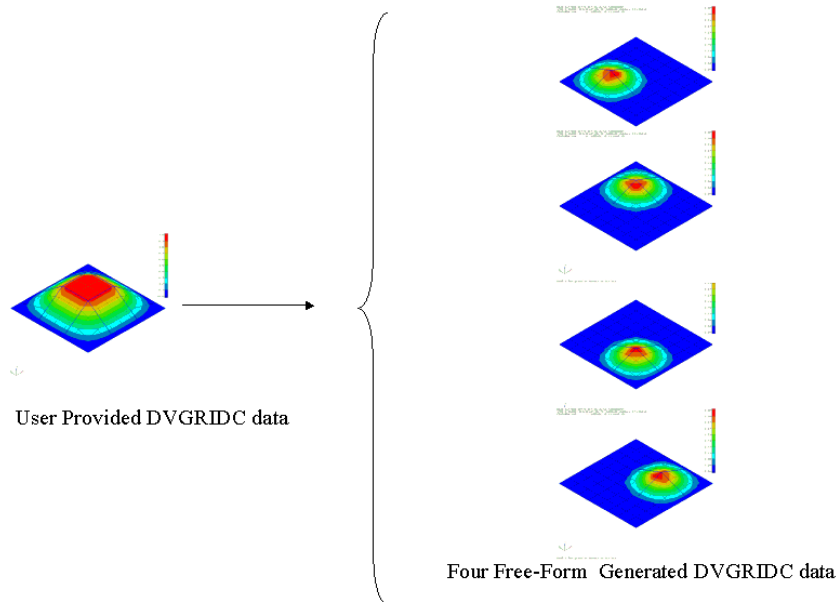
The following data will split all control perturbations associated to design variable 50..:

1	2	3	4	5	6	7	8	9	10
DSHAPE	DVARID	LABEL	SPLIT	FTYPE					
DSHAPE	50	Plate	FREE	DVGRIDC		1.0			

The following figures illustrate the application of freeform with the DVGRIDC option. The four DVGRIDC perturbations are split to reference four independent design variables. The DVGRIDC data generated by those DVGRIDC and the DOMAINS are not split further, but referenced by the four generated variables:.



)



3

.

The values RINIT, MAXPERT and SCALE can also be use with FTYPE=DVGRIDC.

3.6.5 Defining Fabrication (Symmetry) Constraints for Freeform

Each freeform region can optionally be designed using different sets of fabrication constraints. The fabrication constraints are defined using the "SYM" continuation line of **DSHAPE**. Fabrication constraints can only be applied to freeform perturbation (SPLIT=FREE).

The basic format of DSHAPE with fabrication constraints is:

1	2	3	4	5	6	7	8	9	10
DSHAPE	DVARID	LABEL	SPLIT	FTYPE					
+	"SYM"	CID	TYPE1	TYPE2	TYPE3	n	SYMTOL		

Symmetry Planes

The fabrication constraints are defined using symmetry planes. The symmetry planes are defined using existing coordinate systems. In the DSHAPE "SYM" continuation line, CID corresponds to a coordinate system identification number.

A plane of symmetry is internally built using the axes of the selected coordinate system. In other words, The XY symmetry plane is the plane containing the X and Y axes of the coordinate system. The YZ symmetry plane is the plane containing the Y and Z axes of the coordinate system. Finally, the ZX symmetry plane is the plane containing the Z and X axes of the coordinate system.

Defining the Symmetry Fabrication Constraints

Up to three fabrication constraints can be defined with the TYPE1, TYPE2 and TYPE3 fields in the DSHAPE entry. Five basic varieties of fabrication constraints are currently available (mirror, cyclic, axisymmetry, extrusion or uniform) for each of the three directions to make a total of 16 types of fabrication constraints:

TYPE	Description of Fabrication Constraints
MXZ	Mirror symmetry with respect to the MXZ plane
MYZ	Mirror symmetry with respect to the MYZ plane
MZX	Mirror symmetry with respect to the MZX plane
CX	Cyclic symmetry about the X axis (n>0) or Axisymmetry about the X axis (n=0)
CY	Cyclic symmetry about the Y axis (n>0) or Axisymmetry about the Y axis (n=0)

CZ	Cyclic symmetry about the Z axis ($n>0$) or Axisymmetry about the Z axis ($n=0$)
EX	Extrusion along the X axis
EY	Extrusion along the Y axis
EZ	Extrusion along the Z axis
UXY	Uniform with respect to the XY plane
UYZ	Uniform with respect to the YZ pane
UZX	Uniform with respect to the ZX plane
UXYZ	Uniform in all directions

When using cyclic symmetries it is necessary to specify the number of cyclic symmetry repetitions, n . Field 7 is used for that purpose.

The SYMTOL field is used to defined the tolerance associated to finding symmetric grids.

Example: Freeform with mirror symmetry:

1	2	3	4	5	6	7	8	9	10
DSHAPE	70	LABEL	FREE						
+	SYM	1	MX						

Combining Fabrication Constraints

Sometimes it is necessary to impose multiple fabrication constraints on a given freeform region simultaneously. Up to three fabrication constraints can be used per region. However, not all types can be mixed together.

Here is a list of the possible combinations in a single freeform region:

1. Two or three mirror symmetry constraints
2. One cyclic or axisymmetry constraint can be mixed with one mirror symmetry as long as the axis of cyclic symmetry is normal to the plane of mirror symmetry.
3. One extrusion constraint can be mixed with one or two mirror symmetry constraints, as long as the extrusion direction is parallel to the plane(s) of mirror symmetry.
4. One extrusion constraint can be mixed with one cyclic or axisymmetry fabrication constraint, as long as the extrusion direction and the cyclic or axisymmetry axis are the same.
5. One uniform (planar) constraint can be mixed with one mirror symmetry as long as the plane of the uniform constraint is parallel to the plane of mirror symmetry.

Shape and Sizing Design Capabilities

More precisely, here are the list of possible combinations:

If i,j,k are cyclic permutations of X,Y and Z

- 1. Mij and/or Mjk and/or Mki
- 2. Ci and/or Mjk
- 3. Ei and/or Mij and/or Mki
- 4. Ei and/or Ci
- 5. Ujk and/or Mjk

3

Examples: Mixing fabrication constraints

Here are possible combinations:

(MXY, MYZ AND MZX)
(CX AND MYZ) or (CY AND MZX) or (CZ AND MXY)
(EX, MXY AND MZX) or (EY, MYZ AND MXY) or (EZ, MZX AND MYZ)
(EX AND CX) or (EY AND CY) or (EZ AND CZ)
(UYZ AND MYZ) or (UZX AND MZX) or (UXY AND MXY)

Example: Freeform with triple mirror symmetry:

1	2	3	4	5	6	7	8	9	10
DSHAPE	80	LABEL	FREE						
+	SYM	1	MXY	MYZ	MZX				

3.6.6 Coarse Freeform

Freeform optimization can also be used as group by group of grid shape optimization capability. The groups are created by proximity.

Format:

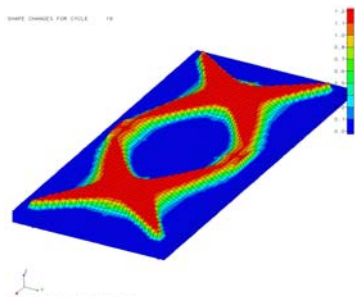
1	2	3	4	5	6	7	8	9	10
DSHAPE	DVARID	LABEL	SPLIT	FTYPE					
+	"COARSE"	CTYPE	DIMEN1						

The advantage of using COARSE is to reduce the number of design variables. This speeds up the sensitivity calculations and the optimization process. Coarse sometimes reduces variability in the design, however reducing variability can often be achieved more effectively using the GRIDFR option described in the next section.

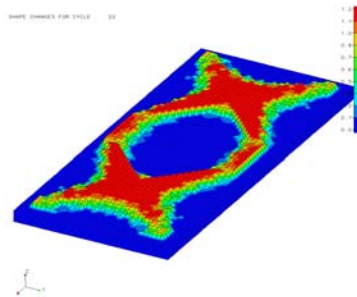
3

Example:

The following example shows two answers to demonstrate the use of COARSE. The figure on the left corresponds to a case where the COARSE option is not used. The figure on the right correspond to a case where COARSE is used. The number of design variables used in the first case is 608 while in the second is 109.



Freeform Results without COARSE



Freeform Results with COARSE

3.6.7 Variability Control - Using the GRIDFR Option in Freeform

The optional GRIDFR continuation line in DSHAPE controls the fraction of grids in a freeform region that are allowed to be moved in the final design. A value of 0.0 would allow no movement, while a value of 1.0 would allow all grids to move to their upper or lower bound.

The basic format of DSHAPE with GRIDFR constraints is:

1	2	3	4	5	6	7	8	9	10
DSHAPE	DVARID	LABEL	SPLIT	FTYPE	MAXPERT	RINIT	SCALE		
+	"GRIDFR"	FRACT	BTYPE	GFRTOL					

The GRIDFR label indicates that variability controls are used.

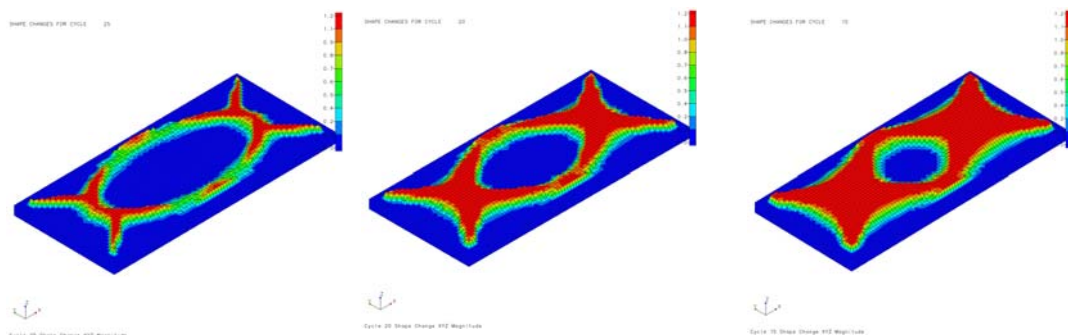
FRACT is a real number between 0.0 and 1.0. A value of 0.5, for example, indicates that only 50% of the grids can move.

BTYPE can be UPPER, LOWER, BOTH or blank. UPPER is to get an upper bound bead fraction constraint (i.e., at most that many grids may move). LOWER is to get a lower bound bead fraction constraint (i.e., at least that many grids must move). BOTH is to get a lower and an upper bound bead fraction constraint (i.e., exactly that many grids shall move). Default is UPPER.

GFRTOL is a small number defining a tolerance for accepting the bead fraction constraint as satisfied. Default 0.005.

Example (Variability control in one freeform regions):

In the following example, three different FRACT values are used 0.25, 0.45 and 0.65 to show the impact of this parameter. BTYPE=UPPER is used in all of them. The structure is subject to a torsional loads .



3.6.8 Freeform Design Variables listed in DSELECT and/or DRESP3 Built- in functions

A user created design variable that is associated to freeform (a design variable that is listed in DSHAPE and DVGRID or DVGRIDC) can be listed in a **DSELECT** entry and/or in a **DRESP3** built-in equation. In this case, the program will add to the DSELECT and/or DRESP3 design variable list all the automatically generated design variables associated to the corresponding freeform.

In a single DRESP3 or DSELECT it is possible to list multiple design variable corresponding to multiple freeform regions. As shown in the next example, this capability can be used to limit how much the design variables from two or more freeform region move. This in turn limits how much the freeform grids can move.

Example (Variability control on multiple freeform regions):

In the following example, the movement of three user created design variable plus all the associated design variables that are created by the program for freeform will be averaged and constrained to move up to 0.5.

1	2	3	4	5	6	7	8	9	10
DSHAPE	10	face1	FREE	DVGRID					
DSHAPE	20	face2	FREE	DVGRID					
DSHAPE	30	face3	FREE	DVGRID					
DSELECT	1100	faces123	0.5	UPPER	AVG				
+	DVAR	10	20	30					

This allows to control global variability. Incontrast, the GRIDFR option in a single freeform entry allows to control local variability or the variability in the given freeform region.

3.7 Topography Optimization

The **DTGRID** data provides a way to automatically generate perturbation vectors and design variables.

The information specified by a DTGRID entry is as follows:

1. Select a designable region
2. Select if the boundary grids of the designable region are designable or not
3. Select basic dimension of the desirable "bead" pattern: maximum height, minimum width and transition distance
4. Optionally, include fabrication constraints such as mirror, cyclic, axisymmetry and/or extrusion.
5. Optionally, modify the default values for the automatically generated design variable bounds and/or move limits
6. Optionally, select a perturbation direction and scale factor. By default the perturbation vectors are created normal to the designable region
7. Optionally, select certain edge grids to be included in the design or not
8. Optionally, select certain interior grids to be excluded from the topography region
9. Optionally, select a fraction of the grids in a topography region that are allowed to move in

The topography perturbation vectors can be used with any other shape and/or sizing design data.

3.7.1 Topography Designable Elements

GENESIS can topographically design all grid referenced by the following elements:

TOPOGRAPHY DESIGNABLE ELEMENTS	
Element	Property
CTRIA3 CQUAD4 CTRIA6 CQUAD8	PSHELL/PCOMP/PCOMPG/ PSKIN

Notes

Skin elements, such as CQUAD4->PSKIN, are non-structural elements. Skin elements are typically created to cover the external surface of solid element meshes. Skin elements can be used with topography data to easily design the solid elements that they cover.

3.7.2 Topography Designable Region Selection

To select a designable region the user needs to specify a group of elements. This is done by selecting a PSHELL, PCOMP, PSKIN, PROPSET or ESET id. All elements referencing the selected property (or set of properties) or all elements in the selected element set will become the designable region.

The basic format for DTGRID is:

Format:

1	2	3	4	5	6	7	8	9	10
DTGRID	ID		PTYPE	PID					
+	"SHAPE"	STYPE	DIMEN1	DIMEN2	DIMEN3				

The ID is a unique identification number to identify the DTGRID data. The ID is only used by the program to report errors in the input data.

PTYPE is a indicator of the region type. IF PTYPE is PSHELL, PCOMP or PSKIN, then PID is the property identification number. IF PTYPE is PROPSET, then PID corresponds to a PROPSET identification number. All elements that reference the property id (or any property id in the set) will be designed. If PTYPE is ESET, then PID corresponds to a set identification of an element set defined by the SET solution control command.

3.7.3 Topography Basic Shape

In a DTGRID entry, a continuation line with the SHAPE keyword is required. Data on this continuation line specifies the basic shape pattern that each design variable controls. Three basic shape patterns are available: CONE, LINE or UNIF. Real parameters control the dimensions of the basic shapes for each type as follows:

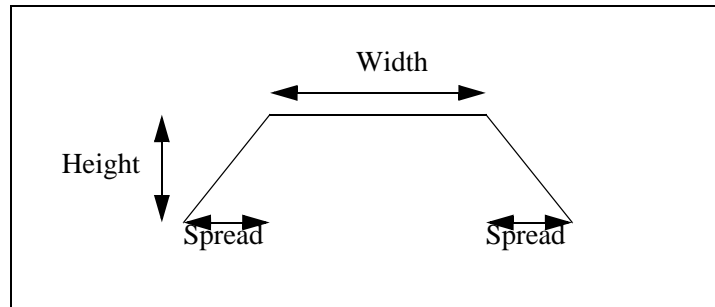
- STYPE=CONE

DIMEN1 = Maximum height. This puts a bound on how much the grids will be able to move in the direction of the perturbation.

DIMEN2 = Minimum width. This controls a minimum width for the case where grids reach the maximum height. This parameter is internally used to create the independent design variables needed to solve the problem. In general, the larger the width, the smaller the number of generated independent design variables.

DIMEN3 = Spread. This controls the influence of each design variable on its neighbors. For the option EDGEM=NOEDGE it also allows a transition zone between designable grids and the boundary of the designable region. In general, the larger the spread, the larger the transition zones.

:



- STYPE=LINE

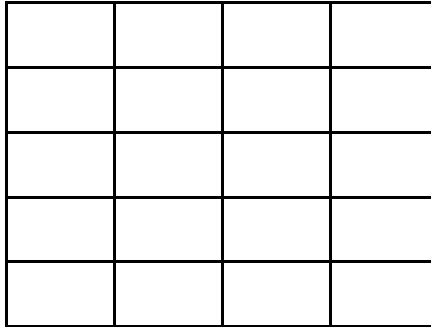
DIMEN1 = Maximum height. This puts a bound on how much the grids will be able to move in the direction of the perturbation.

- STYPE=UNIF

DIMEN1 = Maximum height. This puts a bound on how much the grids will be able to move in the direction of the perturbation. In this case, all designable grids in the topography region will move the same amount.

Example 1

Consider the following mesh that contains 30 GRIDS (12 interior grids) and 20 CQUAD4 element. Consider that each element references the same property defined in PSHELL 100:



If the topography task is to design each of the 12 interior grids, the following data could be created:

	1	2	3	4	5	6	7	8	9	10
DTGRID	1			PSHELL	100	NOEDGE				
+	SHAPE	CONE	2.5							

In DTGRID 1, the NOEDGE option is used for EDGEM. Therefore *GENESIS* will not design the grids on the edges. Since DIMEN1=2.5, *GENESIS* will optimize the plate moving the grids normal to the plane, up to 2.5 units in the positive or negative direction of the perturbation.

If the topography task is to design each of the 30 interior grids, the following data could be created:

	1	2	3	4	5	6	7	8	9	10
DTGRID	2			PSHELL	100	EDGE				
+	SHAPE	CONE	2.5							

In DTGRID 2 EDGEM=EDGE, therefore *GENESIS* will include the grids at the edge in the design. Since DIMEN1=2.5 *GENESIS* will optimize the structures moving the grids up to 2.5 units in the positive or negative direction of the perturbation.

3.7.4 Defining Fabrication (Symmetry) Constraints for Topography

Each topography region can optionally be designed using different sets of fabrication constraints. The fabrication constraints are defined using the "SYM" continuation line in **DTGRID**.

The basic format of DTGRID with fabrication constraints is:

1	2	3	4	5	6	7	8	9	10
DTGRID	ID	LABEL	PTYPE	PID	EDGEM	PERTM	SOLM		
+	"SHAPE"	STYPE	DIMEN1	DIMEN2	DIMEN3				
+	"SYM"	CID	TYPE1	TYPE2	TYPE3	n	SYMTOL		

3

Symmetry Planes

The fabrication constraints are defined using symmetry planes. The symmetry planes are defined using existing coordinate systems. In the DTGRID "SYM" continuation line, CID corresponds to a coordinate system identification number.

A plane of symmetry is internally built using the axes of the selected coordinate system. In other words, the XY symmetry plane is the plane containing the X and Y axes of the coordinate system. The YZ symmetry plane is the plane containing the Y and Z axes of the coordinate system. Finally, the ZX symmetry plane is the plane containing the Z and X axes of the coordinate system.

Defining the Symmetry Fabrication Constraints

Up to three fabrication constraints can be defined with the TYPE1, TYPE2 and TYPE3 fields in the DTGRID entry. Five basic varieties of fabrication constraints are currently available (mirror, cyclic, axisymmetry, extrusion and uniform) for each of the three directions to make a total of 16 types of fabrication constraints:

TYPE	Description of Fabrication Constraints
MXY	Mirror symmetry with respect to the XY plane
MYZ	Mirror symmetry with respect to the YZ pane
MZX	Mirror symmetry with respect to the ZX plane
CX	Cyclic symmetry about the X axis (n>0) or Axisymmetry about the X axis (n=0)

CY	Cyclic symmetry about the Y axis (n>0) or Axisymmetry about the Y axis (n=0)
CZ	Cyclic symmetry about the Z axis (n>0) or Axisymmetry about the Z axis (n=0)
EX	Extrusion along the X axis
EY	Extrusion along the Y axis
EZ	Extrusion along the Z axis
UXY	Uniform in planes parallel to the XY plane
UYZ	Uniform in planes parallel to the YZ pane
UZX	Uniform in planes parallel to the ZX plane
UXYZ	Uniform in all directions

When using cyclic symmetries it is necessary to specify the number of cyclic symmetry repetitions, n. Field 7 is used for that purpose.

The SYMTOL field is used to defined the tolerance associated to finding symmetric elements.

Example: Topography with mirror symmetry:

1	2	3	4	5	6	7	8	9	10
DTGRID	100	LABEL	PSHELL	100					
+	"SHAPE"	CONE	10.0	2.0					
+	SYM	1	MX						

Combining Fabrication Constraints

Sometimes it is necessary to impose multiple fabrication constraints on a given topography region simultaneously. Up to three fabrication constraints can be used per region. However, not all types can be mixed together.

Here is a list of the possible combinations in a single topography region:

- Two or three mirror symmetry constraints
- One cyclic or axisymmetry constraint can be mixed with one mirror symmetry as long as the axis of cyclic symmetry is normal to the plane of mirror symmetry.
- One extrusion constraint can be mixed with one or two mirror symmetry constraints, as long as the extrusion direction is parallel to the plane(s) of mirror symmetry.

- One extrusion constraint can be mixed with one cyclic or axisymmetry fabrication constraint, as long as the extrusion direction and the cyclic or axisymmetry axis are the same.

More precisely, here are the list of possible combinations:

If i,j,k are cyclic permutations of X,Y and Z

- Mij and/or Mjk and/or Mki
- Ci and/or Mjk
- Ei and/or Mij and/or Mki
- Ei and/or Ci
- Ujk and/or Mjk

3

Examples: Mixing fabrication constraints

Here are possible combinations:

(MXY, MYZ AND MZX)

(CX AND MYZ) or (CY AND MZX) or (CZ AND MXY)

(EX, MXY AND MZX) or (EY, MYZ AND MXY) or (EZ, MZX AND MYZ)

(EX AND CX) or (EY AND CY) or (EZ AND CZ)

(UYZ AND MYZ) or (UZX AND MZX) or (UXY AND MXY)

Example: topography with triple mirror symmetry:

1	2	3	4	5	6	7	8	9	10
DTGRID	100	LABEL	PSHELL	100					
+	"SHAPE"	CONE	10.0	2.0					
+	"SYM"	1	MXY	MYZ	MZX				

3.7.5 Topography Design Variable

The program automatically generates design variables for topography. However, the user can control what is generated. By default the following data is provided:

INIT = 0.001 if RINIT is -1.0 or 0.0 if RINIT \geq 0.0

LB = -1.0

UB = 1.0

DELX = 0.001

DXMIN = 0.2

The basic format for DTGRID with DVAR data is:

1	2	3	4	5	6	7	8	9	10
DTGRID	ID		PTYPE	PID				RINIT	
+	"SHAPE"	STYPE	DIMEN1	DIMEN2	DIMEN3				
+	"DVAR"	INIT	LB	UB	DELX	DXMIN			+

The INIT value controls how the initial value of the generated design variables is set. A value of 0.0 would be desired to make the initial shape in the first design cycle identical to the user provide initial shape. However, this would cause the program to calculate zero gradients for areas where the designable region is flat, and no optimization progress would be made. When RINIT is -1.0 (the default), each design variables' initial value is set to an internal random value that is scaled by INIT. In case the program can not move the design variables (as evidenced by stopping with soft convergence in the first design cycle), then the INIT value can be increased to a higher value than the default value (0.001), for example to 0.01.

If topography is used on a shell skin of a solid element part, then the gradients will be non-zero even where the designable region is flat and the design variable is zero. In this case, it may be desirable to start with no randomness. The RINIT value can be used to control the randomness introduced into the initial value. If RINIT is 0.0, then no initial randomness is used, and the INIT value is used as the design variable's initial value. If RINIT > 0.0, then it defines the relative range about INIT from which random initial values are selected.

In the case where it is desired to only allow grids to move to one side of the surface, use LB=0.0 or UB=0.0 as appropriate.

If the mesh gets distorted, especially in early stages, the parameters, DELX and DXMIN, can be used to reduce the move limits associated with the internally created design variables. These parameters alone may only postpone the problem, rather than cure it. If that is the case, the value DIMEN1 may need to be reduced. The COEF parameter described with the PERT keyword also helps to reduce distortions in early design cycles.

3.7.6 Topography Perturbation Vectors

The program automatically generates perturbation vectors for topography. By default, all the perturbations are created normal to the designable surface. However, the user can control the direction of the generated perturbations, if desired. The user can also control a scale factor that will help speed up or slow down the optimization process.

User defined Perturbations

The basic format for DTGRID with PERT data is:

1	2	3	4	5	6	7	8	9	10
DTGRID	ID		PTYPE	PID					
+	"SHAPE"	STYPE	DIMEN1	DIMEN2	DIMEN3				
+	"PERT"	GID	CID	COEFF	PX/G1	PY/G2	PZ/G3		

There are four ways to provide a unique perturbation direction:

1. Picking one GRID and let the program use the normal at that GRID as the unique direction. The selected GRID must belong to the topography region.

In the following example, the normal at GRID 5 will be used as the direction for all perturbations.

1	2	3	4	5	6	7	8	9	10
DTGRID	1		PSHELL	100	NOEDGE				
+	SHAPE	CONE	2.5						
+	PERT	5							

2. Explicitly give the direction on a given coordinate system

In the following example, the direction is given by (0.0, 0.0, 1.0) in the coordinate system 100 located at grid GID. (If the coordinate system is rectangular, then GID maybe left blank. The selected grid GID in this case does not need to belong to the topography region.)

1	2	3	4	5	6	7	8	9	10
DTGRID	1		PSHELL	100	NOEDGE				
+	SHAPE	CONE	2.5						
+	PERT	6	100		0.0	0.0	1.0		

Shape and Sizing Design Capabilities

- Picking 2 GRIDs and let the program use the direction from the first to the second

In the following example, the direction is calculated by creating a vector that points from GRID 51 to GRID 53

	1	2	3	4	5	6	7	8	9	10
DTGRID	1			PSHELL	100	NOEDGE				
+	SHAPE	CONE		2.5						
+	PERT					51	53			

- Picking 3 non-collinear GRIDs and let the program use the normal to the plane defined by the three

In the following example, the direction is calculated to be normal to the plane defined by grids: 1,5,7.

	1	2	3	4	5	6	7	8	9	10
DTGRID	1			PSHELL	100	NOEDGE				
+	SHAPE	CONE		2.5						
+	PERT					1	5	7		

Scale Factor

The coefficient defined in field 5 is used to scale all perturbation in the topography region. This scale factor should be used in the cases where optimization moves too fast or too slow.

In the following example, COEFF has being change to 0.5.

	1	2	3	4	5	6	7	8	9	10
DTGRID	100			PSHELL	100	NOEDGE				
+	SHAPE	CONE		2.5						
+	PERT				0.5					

This scale factor is also used to divide the bounds on the design variable. So in the previous examples, all bounds would be scaled by 2.0 (1.0/0.5).

3.7.7 Adding Boundary Grids to the Topography Designable Region

When the NOEDGE options is used for EDGEM, by default the program will take out all the grids on the edges out of the designable region. There are two method available to add selected edge grids back to the designable region:

1. Adding an explicit list of edge grids (DGRID)

In the following example, GRIDs 100, 101 and 102 that belongs to the edge will be designed.

1	2	3	4	5	6	7	8	9	10
DTGRID	5		PSHELL	100	NOEDGE				
+	SHAPE	CONE	2.5						
+	DGRID	100	101	102					

2. Adding an existing SET of edge grids (DGSET).

In the following example, all grids belonging to grid set 1000 (defined by the SET solution control command) will be included to the topography region.

1	2	3	4	5	6	7	8	9	10
DTGRID	6		PSHELL	100	NOEDGE				
+	SHAPE	CONE	2.5						
+	DGSET	1000							

Notes

Grids that are listed using DGRID or DGSET but do not belong to an edge of the topography region will be ignored.

Both methods can be used simultaneously.

3.7.8 Excluding Grids of a Topography Designable Region

There are two ways to exclude grids from a designable region:

1. Adding an explicit list of non designable grids (NDGRID).

In the following example, GRIDS 200, 201 and 202 will not be designed.

1	2	3	4	5	6	7	8	9	10
DTGRID	1		PSHELL	100	NOEDGE				
+	SHAPE	CONE	2.5						
+	NDGRID	200	201	202					

2. Adding an existing SET of non designable grids (NGSET).

In the following example, all grid belonging to grid set 1000 (defined by the SET solution control command) are excluded from the topography region.

1	2	3	4	5	6	7	8	9	10
DTGRID	1		PSHELL	100	NOEDGE				
+	SHAPE	CONE	2.5						
+	NDGSET	1000							

Notes

GRIDs excluded from one topography region may be designed by a different DTGRID entry.

Both methods can be used simultaneously.

If a GRID is listed to be designanble and also listed to be non-designable then the program will not design the GRID location.

3.7.9 Bead Fraction Control - Using the BEADFR Option in Topography

The optional BEADFR continuation line in DTGRID controls the fraction of grids in a topography region that are allowed to be moved in the final design. A value of 0.0 would allow no movement, while a value of 1.0 would allow all grids to move to their upper or lower bound.

The basic format of DTGRID with BEADFR constraints is:

1	2	3	4	5	6	7	8	9	10
DTGRID	ID	LABEL	PTYPE	PID	EDGEM	PERTM	SOLM		
+	"SHAPE"	STYPE	DIMEN1	DIMEN2	DIMEN3				
+	"BEADFR"	FRACT	BTYPE	TOL					

The BEADFR label indicates that bead fraction controls are used.

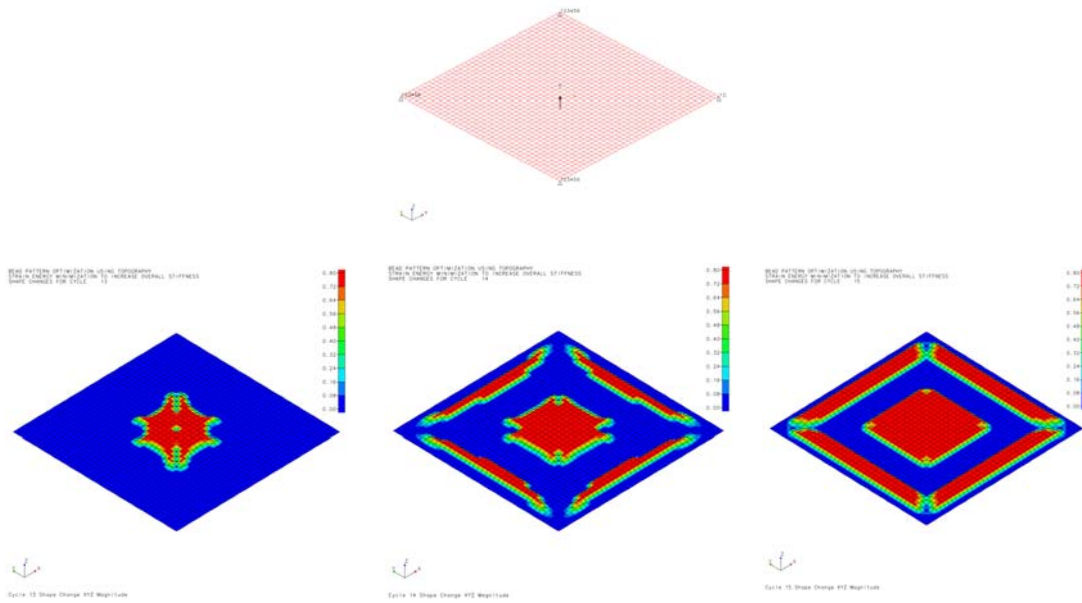
FRACT is a real number between 0.0 and 1.0. A value of 0.5, for example, indicates that only 50% of the grids can move.

BTYPE can be UPPER, LOWER, BOTH or blank. Upper is to get an upper bound bead fraction constraint (i.e., at most that many grids may move). Lower is to get a lower bound bead fraction constraint (i.e., at least that many grids must move). Both is to get a lower and an upper bound bead fraction constraint (i.e., exactly that many grids shall move). Default is UPPER.

TOL is a small number defining a tolerance for accepting the bead fraction constraint as satisfied. Default 0.005.

Shape and Sizing Design Capabilities

Example: In the following example, 3 difference FRACT values are used 0.1, 0.3 and 0.5. BTYPE=UPPER is used in all of them. The structure is subject to a point load in the center.



3.7.10 Print Equivalent DVAR, DLINK and DVGRID data for Topography Optimization

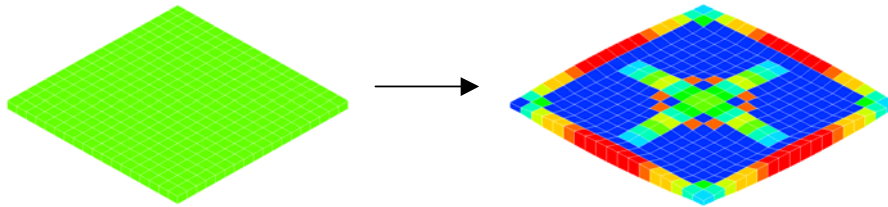
The solution control command DVGRID=PRINT can be used to print the automatically generated DVAR, DVGRID and DLINK data.

3.8 Topometry Optimization

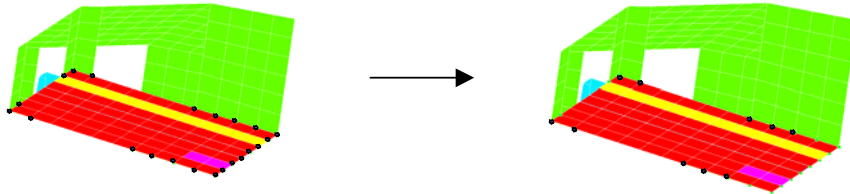
Topometry optimization is an element by element sizing optimization capability. It allows to find an optimal distribution of any property that can be size optimized. This capability is unique to *GENESIS*, and was added to increase the designable space freedom for problems where the user has flexibility for design changes.

Applications for topometry optimization can be numerous. Here are some simple examples used to illustrate the basic capability of this type of optimization.

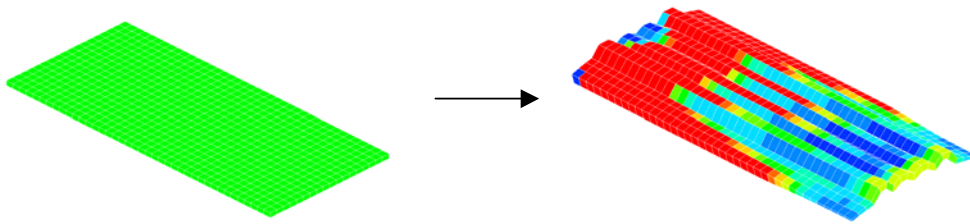
1. Topometry optimization can be used to find the thickness distribution of plates. The following figure shows the initial and optimized design of a simply supported plate subject to a vertical load in the center



2. Topometry optimization also allows to find the best elements to keep from a pool of elements. In particular it can be use to find the best spot weld elements to keep from a list of candidate spot welds as is show in the next figure:.



3. Topometry optimization can be used together with any shape, topography or sizing data. The following figure shows a result of using simultaneously DOMAIN, DTGRID and DSPLIT entries in one run. In this example, the DOMAIN allows to design the global curvature of the plate, the DTGRID data allows to design the bead patterns in the plate, the DSPLIT allows to find thickness distribution. The thickness distribution is shown with different colors (red for thicker areas, blue for thinner areas).



3

The problems shown above are described in the Design example manual. The input data corresponding to these problems are D058.dat, D059.dat and D060.dat and are provided with the program distribution CD.

Fabrication Constraints

Topometry optimization data allows to add fabrication constraints. Currently, the following fabrication constraints are allowed: Mirror symmetry, cyclic symmetry and extrusion.

Coarse Topometry

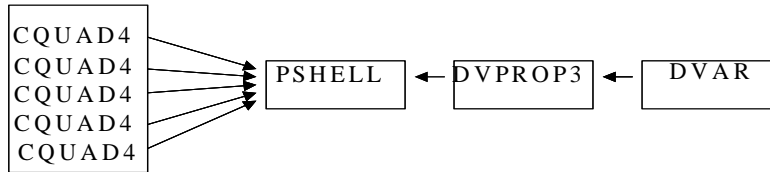
Topometry optimization can also be used as group by group sizing optimization capability. The groups are internally created based on input parameters. The user, for example, specifies the number of elements in each group as n . The groups are created so elements on it are close to each other. Coarse topometry is typically used to reduce the number of design variables to reduce the computational time spent in the sensitivity calculations and on the optimization module.

3.8.1 How to Set Up Topometry Optimization

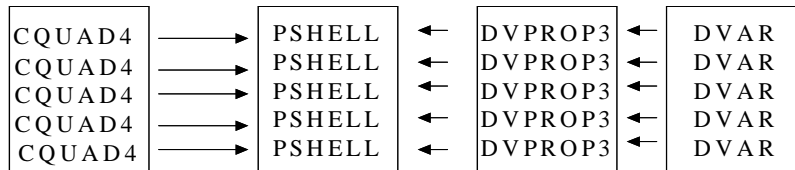
Topometry optimization is controlled by the **DSPLIT** data entry.

To use the topometry optimization capability it is required first to create the relevant sizing optimization data. This is done using the design variable entries (**DVAR/DLINK**) with the standard *GENESIS* design variable to property relationships **DVPROP1**, **DVPROP2**, **DVPROP3** and/or **DVPROP4**.

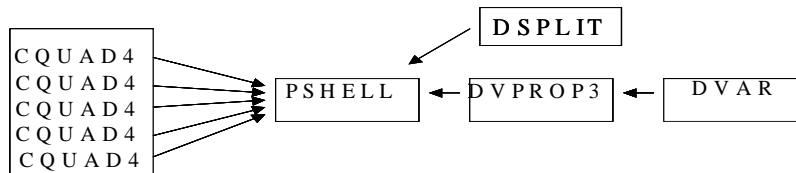
For example, basic sizing optimization data for a 5 element shell model is shown next:



If the user wanted to perform element-by-element optimization, then each CQUAD4 element would need to point to a different PSHELL entry, with corresponding design data:



The **DSPLIT** data statements allows to automatically generate the above data using the following basic sizing data.



The above data is not only far easier to create, but also preserves the original property IDs that are used not only to reference materials in the analysis data but also can be used by other design data (i.e., **DRESP1**, **DTGRID**, etc.).

Topometry optimization can be used with any sizing optimization data. In other words, any element that can be size optimized can be topometry optimized. The following section lists all the topometry designable elements and their associated property data entries.

Topometry Designable Elements

GENESIS can topometrically design any element that can be size optimized. Following is a list of all elements designable with sizing/topometry optimization:

TOPOMETRICALLY DESIGNABLE ELEMENTS	
Element	Property
CELAS1	PELAS
CVECTOR	PVECTOR
CBUSH	PBUSH
CMASS1	PMASS
CONM3	PCONM3
CDAMP	PDAMP
CVISC	PVISC
CROD	PROD
CBAR	PBAR
CHBDY	PHBDY
CSHEAR	PSHEAR
CTRIA3	PSHELL/PCOMP/PCOMPG
CQUAD4	PSHELL/PCOMP/PCOMPG
CTRIA6	PSHELL/PCOMP/PCOMPG
CQUAD8	PSHELL/PCOMP/PCOMPG
CTRIAX6	PAXIS
K2UU1	PK2UU
M2UU1	PM2UU

In a given input file, all or some of these elements can be simultaneously designed. All other elements, such as solid elements, can be used in the input data, but they can not be topometrically designed. Shape and topography data can be used to design these elements simultaneously. For example, the grid locations and the thickness of a QUAD4 element can be simultaneously designed.

3.8.2 Topometry Design Bulk Data

The key data to perform topometry optimization is the **DSPLIT** data entry. This data entry is used to:

1. Define design regions where the topometry optimization can be applied.
2. Select which design variables associated to the topometry region will be split. By default, all design variables associated to a property are split.

Selecting a Topometry Region

Elements associated to a property that are designed with DVPROP1, DVPROP2, DVPROP3 and/or DVPROP4 data can be selected to be topometrically designed. The DSPLIT data entry is used to select the property ID of the elements that are to be topometrically designed.

The basic format for DSPLIT is:

1	2	3	4	5	6	7	8	9	10
DSPLIT	ID	LABEL	PTYPE	PID					

In the above format, the ID is a unique identification number to identify the DSPLIT data. The ID is used by the program to report errors to the user. The LABEL is used to help the user identify the topometry region. The PTYPE indicates the property type: PAXIS, PROD, PBAR, PBUSH, PSHELL, PCOMP, PELAS, PVECTOR, PSHEAR, PCONM3, PHBDY, PDAMP, PMASS or PVISC, also PROP or blank is allowed. The PID is the property identification number, the default value for PID is ID.

Using default values a region can also be selected using:

1	2	3	4	5	6	7	8	9	10
DSPLIT	PID								

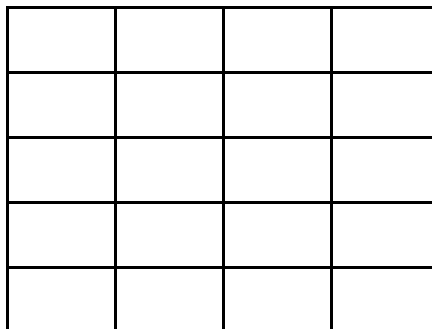
in this case PID and ID will have same value.

Like in any other bulk data, DSPLIT can be placed anywhere in the input data.

Multiple DSPLIT entries are allowed in the input data. But, at most one DSPLIT entry may reference a given property ID.

Example 1:

Consider the following mesh that contains 20 CQUAD4 element and each has the same property defined in PSHELL 100:



The following data could be used to size optimize the structure

1	2	3	4	5	6	7	8	9	10
---	---	---	---	---	---	---	---	---	----

DVPROP3	10	100	SOLID						
+	10								
DVAR	10	THICK	1.0	0.1	3.0				

With the above data the program would design all 20 elements with one design variable. If the design optimization task is to design each of the 20 elements individually, the following data could be created:

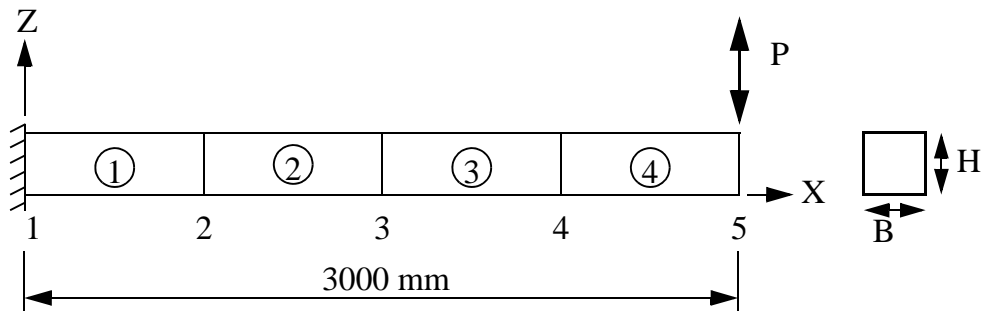
	1	2	3	4	5	6	7	8	9	10
DSPLIT	1	Plate	PSHELL	100						

With this data *GENESIS* will internally create 19 additional design variables. Each of the design variables will have the same information as the provided design variable. Internally, *GENESIS* will also create 19 PSHELL entries and 19 DVPROP3 relationships. The following data would produce same results:

	1	2	3	4	5	6	7	8	9	10
DSPLIT	100									

Example 2:

Consider the following mesh that contains 4 CBAR elements and each has the same property defined in PBAR 107:



The following data could be used to size optimize the bar

	1	2	3	4	5	6	7	8	9	10
DVPROP3	10	107	RECT							
+	1	2								
DVAR	1	B	5.0	0.1	10.0					
DVAR	2	H	1.0	0.1	3.0					

Shape and Sizing Design Capabilities

If the topometry optimization task is to design each of the 4 elements with different width and heights then the following data could be created:

1	2	3	4	5	6	7	8	9	10
DSPLIT	23	B&H	PBAR	107					

With this data *GENESIS* will internally create 6 (3*2) additional design variables, 3 new PBAR entries and 3 new DVPROP2 relationships.

3.8.3 Topometry Design Variables

The program automatically generates design variables for each topometry region. By default, every design variable that designs the property identified on **DSPLIT** will be "split". That is, new independent design variables will be created for every element in the property. However, there may be situations in which the user does not want to split all design variables designing a property. For example, imagine a part with a two-layer composite, where it is desired to design the two layer thicknesses such that the layer 1 thickness is uniform across the whole property, while the layer 2 thickness can vary element-by-element. In this case, it is desired to split the design variable for layer 2, while having the same single design variable for layer 1 control all of the elements.

To handle this situation the DSPLIT entry allows optional continuation lines to either explicitly list which design variables should be split or to explicitly list which variables should not be split.

The format of DSPLIT with topometry variables is:

1	2	3	4	5	6	7	8	9	10
DSPLIT	ID	LABEL	PTYPE	PID					
+	"DVAR"	ID1	ID2	ID3	ID4	ID5	ID6	ID7	ID8
+		ID9	ID10	...					

or alternatively:

1	2	3	4	5	6	7	8	9	10
DSPLIT	ID	LABEL	PTYPE	PID					
+	"NDVAR"	ID1	ID2	ID3	ID4	ID5	ID6	ID7	ID8
+		ID9	ID10	...					

The "DVAR" label indicates that following is a list of variables that are to be split. Only design variables in this list will be split by the DSPLIT entry. Any variable that designs the property that is not listed will not be split.

The "NDVAR" label indicates that following is a list of variables that are **not** to be split. All variables designing the property will be split except those listed in the NDVAR list.

If neither list is given, then the default is to split all variables designing the property.

Assuming that the following sizing design data is given:

1	2	3	4	5	6	7	8	9	10
DVPROP3	10	107	RECT						
+	1	2							
DVAR	1	B	5.0	0.1	10.0				
DVAR	2	H	1.0	0.1	3.0				

Shape and Sizing Design Capabilities

Suppose it is desired that the design variable 2 be split but the design variable 1 not be split. To indicate this, the following data could be used:

1	2	3	4	5	6	7	8	9	10
DSPLIT	10	ONLY_H	PBAR	107					
+	DVAR	2							

Alternatively, the same effect could have been achieved with the following data:.

1	2	3	4	5	6	7	8	9	10
DSPLIT	12	NO_B	PBAR	107					
+	NDVAR	1							

The previous example shows a case where the bar elements associated to property 107 will be designed so each can have a different height ($H=DVAR\ 2$), but all of the elements will have the same width ($B=DVAR\ 1$).

Split Variables

Split variables are all design variables that will be split.

Split variables can be dependent or independent. If a split variable is independent, it can be continuous or discrete. If a split variable is dependent, then it can only be continuous.

Non-split Variables

Non-split variables are design variables designing a topometry region that are not to be split (either they are listed in an NDVAR list or omitted from a DVAR list).

Preservation of Design Variable Implicit Links

If a split design variable is used to design multiple properties or dimensions, then the associated generated variables will preserve that. For example, if a split variable is used to design simultaneously the width and the height of a generic bar property, each of the generated variables will design simultaneously the width and the height of the associated individual elements.

In the following example the design variable 2001 is used to design simultaneously the width and the height of the bar:

1	2	3	4	5	6	7	8	9	10
DVPROP3	10	107	RECT						
+	2001	2001							
DVAR	2001	B&H	5.0	0.1	10.0				
DSPLIT	10	TOPBAR	PBAR	107					

Preservation of DLINK Among Design Variables of Same Region

If some split design variables within a topometry region are linked with a **DLINK** data statement, then the corresponding generated design variables will be linked using new relationship that preserve the links.

For example, the split design variables 2001 and 2002 are linked with DLINK data so that $DVAR\ 2002 = DVAR\ 2001$. The program will split both variables and create corresponding DLINK data so each element will have their generated variables linked. In other words, the program does not only replicate DVAR data but also its associated DLINK data.

	1	2	3	4	5	6	7	8	9	10
DVPROP3	10	107	RECT							
+	2001	2002								
DVAR	2001	B	5.0	0.1	10.0					
DVAR	2002	H	5.0	0.1	10.0					
DLINK	2002			2001	1.0					
DSPLIT	10	TOPBAR	PBAR	107						

Printing Topometry Design Variable

By default, the topometry design variables that are automatically created by the program are not printed in the output file. This is because the output could be too large and the extra design variables are in general not needed. The solution control command **TVAR** = ALL allows to print the internally created design variable in the output file, in this case the internally created design variables are printed together with the user created design variables. The history, *pname.HIS*, file contain all design variables.

Multiple Topometry Regions

Multiple topometry regions can be used in one input file. In fact, there can be as many DSPLIT data statements as designed properties.

However, a split design variable can only be used in one topometry region. An attempt to use a split variable in multiple topometry regions will generate a fatal error message.

GENESIS will not allow to link split variables from different topometry regions. An attempt to do so will generate a fatal error message. One reason for this limitation is that different topometry regions can have different numbers of elements, therefore the number of generated variables in one topometry region could be different than the other. Even if the number were the same, it is not generally clear which generated variables should be linked with which.

Illegal DLINK data

It is not allowed to link split design variables from two different topometry regions.

Consider the following two topometry regions - 201 and 301.

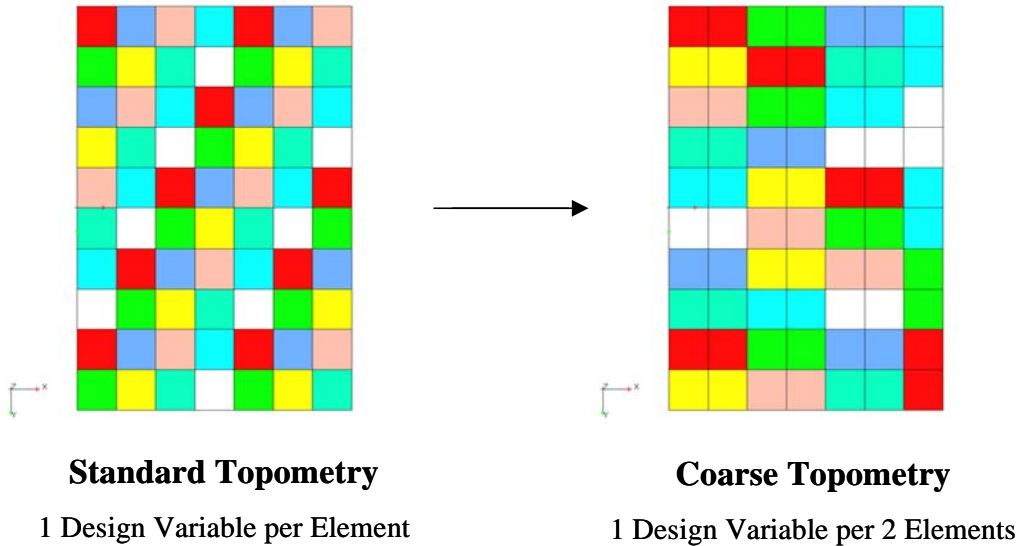
	1	2	3	4	5	6	7	8	9	10
DSPLIT	201	TOPBAR	PBAR	107						
DVPROP3	201	107	RECT							
+	2001	2002								
DVAR	2001	B	5.0	0.1	10.0					
DVAR	2002	H	5.0	0.1	10.0					
DSPLIT	301	TOPBAR	PBAR	108						
DVPROP3	301	108	RECT							
+	3001	3002								
DVAR	3001	B	5.0	0.1	10.0					
DVAR	3002	H	5.0	0.1	10.0					

The following DLINK data would be illegal:

DLINK	3001			2001	1.0				
-------	------	--	--	------	-----	--	--	--	--

3.8.4 Coarse Topometry Optimization

Coarse topometry, is a group by group sizing optimization capability. The groups are automatically created by the program based on a user specified number that indicates how many elements should be in a group. The groups are formed using a technique that group elements that touch each other. Coarse topometry, is a compromise between sizing and standard topometry optimization.



3

Motivation to use Coarse Topometry

When topometrically designable areas contain a large number of elements, a standard topometry optimization problem produces a large number of design variables. This translates to a large number of computations that could make the optimization task run very slow. To reduce the number of variables *GENESIS* can lump elements into small groups and design all element in the group with the same set of design variables. The groups can be created using a user selected number of elements. For example, each group can have four elements. With four elements per group, the number of variables is reduced by a factor of four. The user can select any number of elements per group. The number of element gives a trade off between detail and time of solving the problem, the larger the number of elements per group, the less detail obtained by the optimization result, but *GENESIS* can perform each design cycle faster. When picking the number of elements, it is important to keep in mind how big the model is and how much detail is really needed.

Shape and Sizing Design Capabilities

To specify coarse topometry, the DSPLIT entry has a "COARSE" continuation line:

1	2	3	4	5	6	7	8	9	10
DSPLIT	ID	LABEL	PTYPE	PID					
+	"COARSE"	CTYPE	CVALUE1	CVALUE2	CVALUE3				

In the above entry, COARSE is a label that indicates that the following data corresponds to coarsening data. CTYPE specifies the coarsening procedure to use. If CTYPE=MAXELEM, then the coarsening is defined by the maximum number of elements per group. In this case, CVALUE1 is the maximum number of elements per group. The program will try to put CVALUE1 contiguous elements into each group, although some groups may contain fewer elements. Note the maximum element value does not consider the additional grouping due to fabrication symmetries. If CTYPE=LENGTH, then the coarsening is defined by finite-sized boxes. The whole space is divided into a uniform mesh of boxes, each of dimension CVALUE1 by CVALUE2 by CVALUE3. Elements whose centroids are within the same box are grouped together. The boxes are aligned with the axes of the coordinate system identified by CID on the SYM continuation line (which must be present if CTYPE=LENGTH).

Coarse Topometry Example -Grouping 2 elements per variable

1	2	3	4	5	6	7	8	9	10
DSPLIT	201	TOPBAR	PBAR	107					
+	COARSE	MAXELEM	2						

Coarse Topometry Example 2 -Grouping elements into cubic boxes

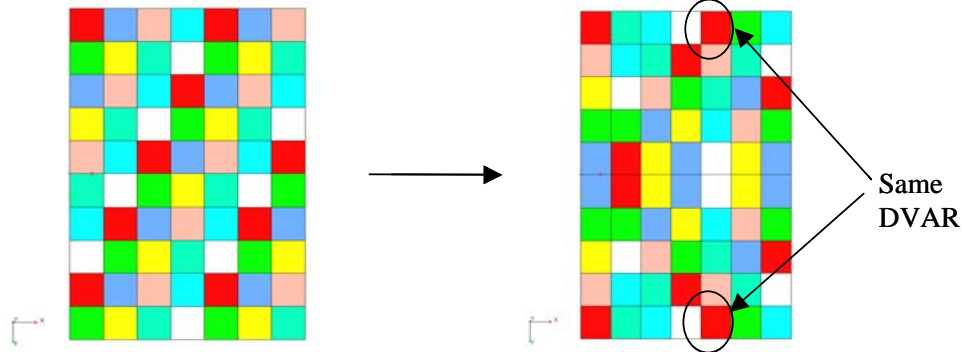
1	2	3	4	5	6	7	8	9	10
DSPLIT	401		PSHELL	109					
+	COARSE	LENGTH	10.0	10.0	10.0				
+	SYM	22							

3.8.5 Defining Fabrication (Symmetry) Constraints for Topometry

With the exception of PK2UU and PM2UU topometry regions, each topometry region can be designed using different sets of fabrication constraints. The fabrication constraints can be defined using the "SYM" continuation line in DSPLIT

The format of DSPLIT with fabrication constraints is:

1	2	3	4	5	6	7	8	9	10
DSPLIT	ID	LABEL	PTYPE	PID					
+	"SYM"	CID	TYPE1	TYPE2	TYPE3	n	SYMTOL	SYMMET	ANGTYPE



Standard Topometry

1 Design Variable per Element

Symmetric Topometry

1 Design Variable per 2 Elements

Symmetry Planes

The fabrication constraints are defined using symmetry planes. The symmetry planes are defined using existing coordinate systems. In the DSPLIT "SYM" continuation line, CID corresponds to a coordinate system identification number.

A plane of symmetries is internally built using the axes of the selected coordinate system. In other words, the XY symmetry plane is the plane containing the X and Y axes of the coordinate system. The YZ symmetry plane is the plane containing the Y and Z axes of the coordinate system. Finally, the ZX symmetry plane is the plane containing the Z and X axes of the coordinate system.

Defining the Symmetry Fabrication Constraints

Up to three fabrication constraints can be defined with the TYPE1, TYPE2 and TYPE3 fields in the DSPLIT entry. Four basic varieties of fabrication constraints are currently available (mirror, cyclic, extrusion and plane uniform) for each of the three directions to make a total of 12 types of fabrication constraints:

TYPE	Description of Fabrication Constraints
MXY	Mirror symmetry with respect to the XY plane
MYZ	Mirror symmetry with respect to the YZ pane
MZX	Mirror symmetry with respect to the ZX plane
CX	Cyclic symmetry about the X axis
CY	Cyclic symmetry about the Y axis
CZ	Cyclic symmetry about the Z axis
EX	Extrusion along the X axis
EY	Extrusion along the Y axis
EZ	Extrusion along the Z axis
UXY	Uniform in planes parallel to the XY plane
UYZ	Uniform in planes parallel to the YZ pane
UZX	Uniform in planes parallel to the ZX plane
UXYZ	Uniform in all directions

When using cyclic symmetries it is necessary to specify the number of cyclic symmetry repetitions, n. Field 7 is used for that purpose.

The SYMTOL field is used to defined the tolerance associated to finding symmetric elements. SYMMET is used to select the method for checking symmetries. There are two optional methods, with SYMMET=1 only the center of the elements are checked. With SYMMET=2, the default, the coordinates of all grids and the center of each element are checked.

Example: Topometry with mirror symmetry:

1	2	3	4	5	6	7	8	9	10
DSPLIT	400		PSHELL	2001					
+	SYM	1	MXY						

Combining Fabrication Constraints

Sometimes it is necessary to impose multiple fabrication constraints on a given topometry region simultaneously. Up to three fabrication constraints can be used per region. However, not all types can be mixed together.

Here is a list of the possible combinations in a single topometry region:

1. Two or three mirror symmetry constraints
2. One cyclic constraint can be mixed with one mirror symmetry as long as the axis of cyclic symmetry is normal to the plane of mirror symmetry.
3. One extrusion constraint can be mixed with one or two mirror symmetry constraints, as long as the extrusion direction is parallel to the plane(s) of mirror symmetry.
4. One extrusion constraint can be mixed with one cyclic symmetry constraint, as long as the extrusion direction and the cyclic axis are the same.
5. One uniform constraint with the mirror constraint about the same plane.

More precisely, here are the list of possible combinations:

If i,j,k are cyclic permutations of X,Y and Z

1. Mij and/or Mjk and/or Mki
2. Ci and/or Mjk
3. Ei and/or Mij and/or Mki
4. Ei and/or Ci
5. Uij and/or Mij

Examples: Mixing fabrication constraints

Here are possible combinations:

(MXY, MYZ AND MZX)

(CX AND MYZ) or (CY AND MZX) or (CZ AND MXY)

(EX, MXY AND MZX) or (EY, MYZ AND MXY) or (EZ, MZX AND MYZ)

(EX AND CX) or (EY AND CY) or (EZ AND CZ)

(UXY AND MXY) or (UYZ AND MYZ) or (UZX AND MZX)

Example: Topometry with triple mirror symmetry:

1	2	3	4	5	6	7	8	9	10
DSPLIT	300		PSHELL	1001					
+	SYM	1	MXY	MYZ	MZX				

Composite Angle treatment

If ANGTYPE is set to ANGSYM (the default) then the angle associated to symmetric elements are negative of each other.

Example: Topometry with simple mirror symmetry and with symmetric angles (ANGSYM)

1	2	3	4	5	6	7	8	9	10
DSPLIT	20		PCOMP	200					
+	SYM	1	MXY						ANGSYM

If ANGTYPE is ANGREP, then the angle associated to symmetric elements are repeated.

Example: Topometry with simple mirror symmetry and repeated angles (ANGREP)

1	2	3	4	5	6	7	8	9	10
DSPLIT	10		PCOMP	100					
+	SYM	1	MXY						ANGREP

The ANGSYM and ANGREP are ignored if the designable elements are not composite.

Important note on composite angle treatment

The angular symmetries are imposed to all elements in the topometry region, regardless of the angles themselves are designed or not.

3.8.6 CoarseTopometry with Fabrication Constraints

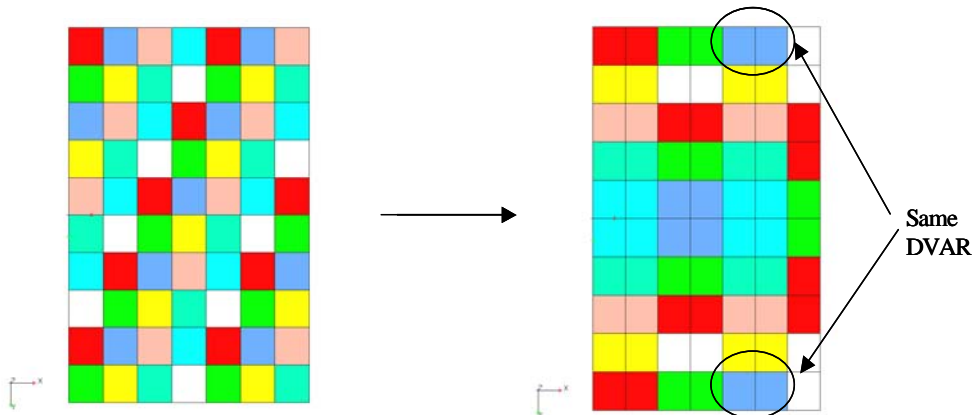
Coarse topometry can be mixed with fabrication constraints.

The format of DSPLIT with fabrication constraints and coarse parameters is:

1	2	3	4	5	6	7	8	9	10
DSPLIT	ID	LABEL	PTYPE	PID					
+	"COARSE"	CTYPE	CVALUE1	CVALUE2	CVALUE3				
+	"SYM"	CID	TYPE1	TYPE2	TYPE3	n	SYMTOL	SYMMET	ANGTYPE

Example;

1	2	3	4	5	6	7	8	9	10
DSPLIT	500		PSHELL	2002					
+	COARSE	MAXELEM	2						
+	SYM	4	MXY						



Standard Topometry

1 Design Variable per Element

Coarse & Symmetric Topometry

1 Design Variable per 2 or 4 Elements

3.8.7 Coarse Topometry with Fabrication Constraints and Selective Design Variables

Coarse topometry can be mixed with fabrication constraints and split design variables.

The full format of DSPLIT with selective split variables, coarse parameters and fabrication constraints is:

1	2	3	4	5	6	7	8	9	10
DSPLIT	ID	LABEL	PTYPE	PID					
+	"DVAR"	ID1	ID2	ID3	ID4	ID5	ID6	ID7	ID8
+		ID9	ID10	...					
+	"COARSE"	CTYPE	CVALUE1	CVALUE2	CVALUE3				
+	"SYM"	CID	TYPE1	TYPE2	TYPE3	n	SYMTOL	SYMMET	ANGTYPE

alternative ;

1	2	3	4	5	6	7	8	9	10
DSPLIT	ID	LABEL	PTYPE	PID					
+	"NDVAR"	ID1	ID2	ID3	ID4	ID5	ID6	ID7	ID8
+		ID9	ID10	...					
+	"COARSE"	CTYPE	CVALUE1	CVALUE2	CVALUE3				
+	"SYM"	CID	TYPE1	TYPE2	TYPE3	n	SYMTOL	SYMMET	ANGTYPE

3.8.8 Extended Topometry Regions

Typically a topometry region references one property and when multiple properties are to be topometrically designed then multiple DSPLIT entries are created. This case works well for most situations, but occasionally we would like to make one topometry region reference multiple property IDs so that symmetry condition listed in the DSPLIT can be enforced across multiple properties.

To handle this situation, the DSPLIT entry allows for optional continuation lines to list all additional properties

The format of DSPLIT with multiple properties is:

1	2	3	4	5	6	7	8	9	10
DSPLIT	ID	LABEL	PTYPE	PID					
+	"PLINK"	PID1	PID2	PID3	PID4	PID5	PID6	PID7	PID8
+		PID9	PID10	...					

Master Property

The property which is listed in the first line, in the field 5 of the DPSLIT entry is called master property.

The master property must have sizing data associated to it.

Slave Properties

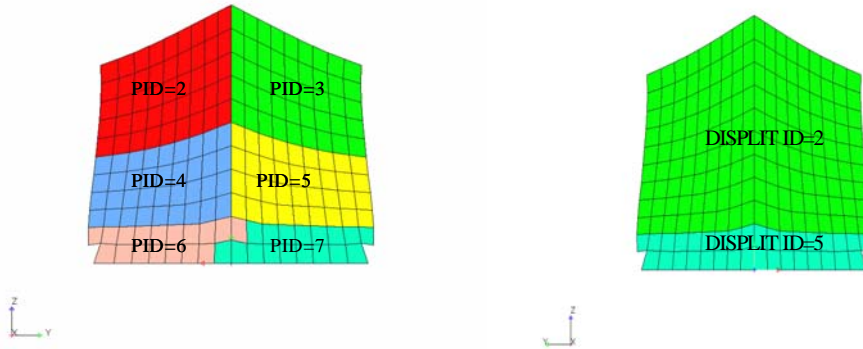
The properties that are listed in the PLINK list are called slave properties.

Slave properties must not have sizing data. Slave properties can only have one master property. Slave properties have to be of the same type as the master property (i.e., a PSHELL master cannot have a PCOMP as a slave property). PCOMP and PCOMPG are considered to be the same type.

The initial values of the properties of a slave property are ignored by *GENESIS*. The property values are inherited from the master property or from the design data generated by topometry.

Example :

Consider a mesh that contains 194 CQUAD4 element associated to 6 PSHELLS; Assume now that you need to create two extended topometry region. The first one will have properties 2, 3, 4 and 5. The second region will include all the elements associated to properties 6 and 7.



The first extended topometry region is:

	1	2	3	4	5	6	7	8	9	10
DSPLIT	2	FourReg	PTYPE	2						
+	PLINK	3	4	5						

The second extended topometry region is:

	1	2	3	4	5	6	7	8	9	10
DSPLIT	5	TwoReg	PTYPE	7						
+	PLINK	6								

The data needed to topometrically design this two regions is

DVPROP3	201	2	SOLID							
+	2001	2002								
DVAR	2001	T2	1.0	0.1	2.0					
DVPROP3	301	7	SOLID							
+	3001									
DVAR	3001	T7	1.0	0.1	2.0					

Note that you should not create design data (DVPROP3) for properties 3, 4, 5 or 6.

3.8.9 Extended Topometry Regions with Global Symmetries

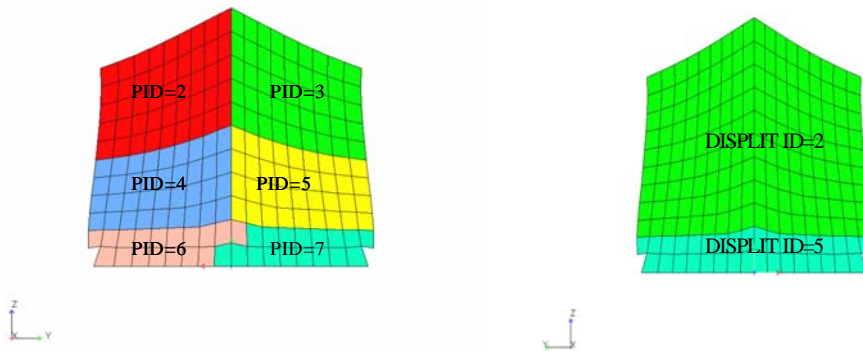
When using extended topometry regions (PLINK option in DSPLIT) you can enforce global symmetries. Global symmetry refers to symmetry that considers elements of more than one property.

The format of DSPLIT with global Symmetry is

1	2	3	4	5	6	7	8	9	10
DSPLIT	ID	LABEL	PTYPE	PID					
+	"PLINK"	PID1	PID2	PID3	PID4	PID5	PID6	PID7	PID8
+		PID9	PID10	...					
+	"SYM"	CID	TYPE1	TYPE2	TYPE3	n	SYMTOL	SYMMET	ANGTYPE

Example :

Consider a mesh that contains 194 CQUAD4 element associated to 6 PSHELLS; Assume now that you need to create two extended topometry region. The first one will have properties 2, 3, 4 and 5. The second region will include all the elements associated to properties 5 and 6.



The first extended topometry region with symmetry condition is:

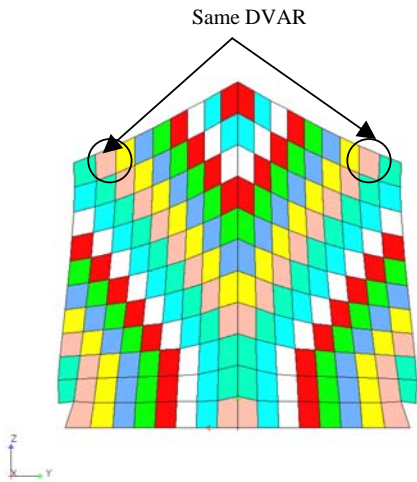
1	2	3	4	5	6	7	8	9	10
DSPLIT	1	FourReg	PTYPE	2					
+	PLINK	3	4	5					
+	SYM	1	MXZ						

Shape and Sizing Design Capabilities

The second extended topometry region with symmetry condition is:

1	2	3	4	5	6	7	8	9	10
DSPLIT	2	TwoReg	PTYPE	7					
+	PLINK	6							
+	SYM	1	MXZ						

3



3.8.10 Extended Topometry with Coarse Parameters, Selective Design Variables , Fabrication Constraints and Global Symmetries.

Extended symmetries can be used with coarse parameters, with fabrication constraints, split design variables and global variables simultaneously.

The full format of DSPLIT is:

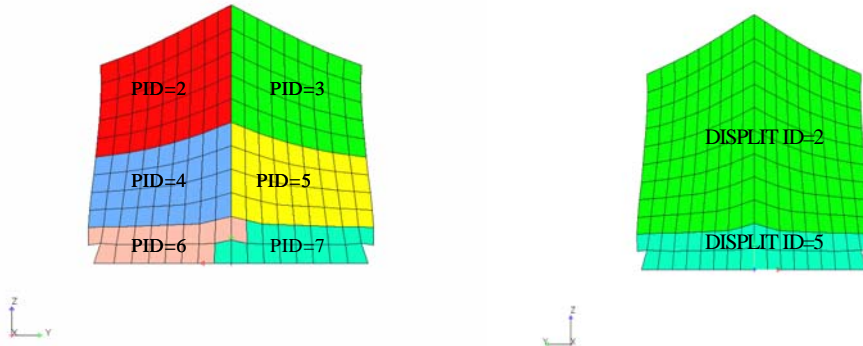
1	2	3	4	5	6	7	8	9	10
DSPLIT	ID	LABEL	PTYPE	PID					
+	"PLINK"	PID1	PID2	PID3	PID4	PID5	PID6	PID7	PID8
+		PID9	PID10	...					
+	"DVAR"	ID1	ID2	ID3	ID4	ID5	ID6	ID7	ID8
+		ID9	ID10	...					
+	"COARSE"	CTYPE	CVALUE1	CVALUE2	CVALUE3				
+	"SYM"	CID	TYPE1	TYPE2	TYPE3	n	SYMTOL	SYMMET	ANGTYPE

alternative ;

1	2	3	4	5	6	7	8	9	10
DSPLIT	ID	LABEL	PTYPE	PID					
+	"PLINK"	PID1	PID2	PID3	PID4	PID5	PID6	PID7	PID8
+		PID9	PID10	...					
+	"NDVAR"	ID1	ID2	ID3	ID4	ID5	ID6	ID7	ID8
+		ID9	ID10	...					
+	"COARSE"	CTYPE	CVALUE1	CVALUE2	CVALUE3				
+	"SYM"	CID	TYPE1	TYPE2	TYPE3	n	SYMTOL	SYMMET	ANGTYPE

Example :

Consider a mesh that contains 194 CQUAD4 element associated to 6 PSHELLS;
Assume now that you need to create two extended topometry region. The first one will have properties 2, 3, 4 and 5. The second region will include all the elements associated to properties 6 and 7.



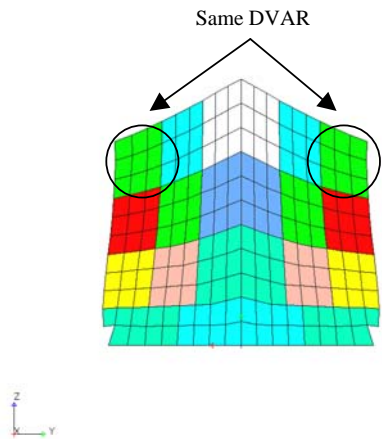
The first extended topometry region with coarse parameters and symmetry condition is:

	1	2	3	4	5	6	7	8	9	10
DSPLIT	1	FourReg	PTYPE	2						
+	PLINK	3	4	5						
+	COARSE	MAXELEM	9							
+	SYM	1	MXZ							

The second extended topometry region with coarse parameters and symmetry condition is:

	1	2	3	4	5	6	7	8	9	10
DSPLIT	2	TwoReg	PTYPE	7						
+	PLINK	6								
+	COARSE	MAXELEM	9							
+	SYM	1	MXZ							

Genesis in this case will use 11 independent design variables. Each design variables controls up to 2*9 elements (the factor 2 comes from symmetry the 9 comes from coarse level).



3

3.8.11 Move Limits in Topometry Optimization

As in regular sizing and shape optimization, *GENESIS* topometry optimization uses an approximate problem to solve the optimization problem more efficiently. In general, the approximations are accurate close to the design variable value at which they were constructed but as the variables move from there, the accuracy decays. This makes it necessary to limit the range of design variable values for which the approximations are used. These limits will be used as temporary bounds in a particular design cycle. At the end of the optimization the move limits should not have affected the results.

The program scales the default move limits of all split design variables by the DOPT parameter **DXFRAC**. In other words,

$$\text{DXMIN (topometry)} = \text{DXFRAC} * \text{DXMIN}$$

$$\text{DELX (topometry)} = \text{DXFRAC} * \text{DELX}$$

The default for DXFRAC = 1.0.

If DXFRAC is changed to 0.5, for example, and DVAR DXMIN = 0.1 and DELX = 0.5 then the program will internally use: DXMIN = 0.5*0.1 = 0.05 and DELX = 0.5*0.5 = 0.25.

In topometry optimization *GENESIS*, by default, does not use property move limits.

3.8.12 Split Responses

GENESIS can use any of the allowed responses in shape, sizing and topography optimization. In other words, **DRESP1**, **DRESP2**, **DRESP3** and **DRESPG**. **DRESPU** can also be used, however, *GENESIS* does not provide a way for the external user program to know which generated design variables are designing which elements. All generated design variables will come after explicitly defined design variables in the **GNUSER** *dvar_file*.

There are some considerations that need to be taken into account when using certain responses with topometry. The following section discusses what occurs in the code when split variables and topometrically designed properties are listed on DRESP1, DRESP2 and DRESP3 data entries.

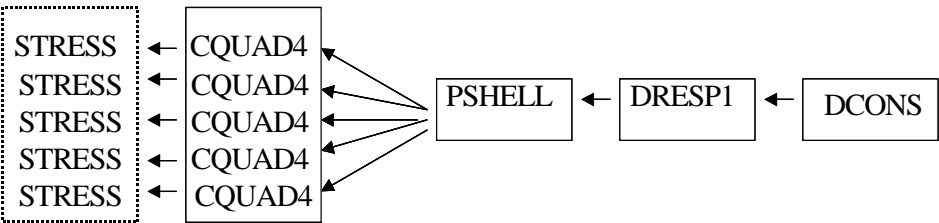
3.8.13 Using DRESP1 with Topometry Data

The format for the DRESP1 data is:

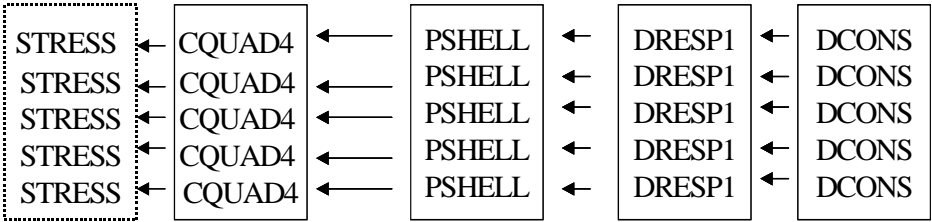
1	2	3	4	5	6	7	8	9	10
DRESP1	ID	LABEL	RTYPE	PTYPE		ATTA	ATT1	ATT1	etc.

The program allows the reference of topometrically designed properties with DRESP1 data, just as it allows reference of any other ordinary property.

Example: Consider the case where the user creates a DRESP1 data that tags stress of a PSHELL that has 5 CQUAD4 elements:.



If the PSHELL data is split as a topometry region, then program will internally create data equivalent to the following:



The above DRESP1 is termed a “split DRESP1”.

Is important to understand this in order to fully understand what GENESIS does when a split DRESP1 is referenced by DRESP2 data. This special case is discussed next.

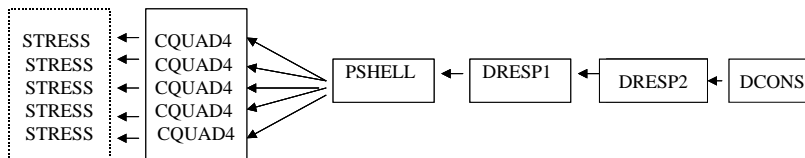
3.8.14 Using DRESP2 with Split DRESP1 and/or Split Design Variables

The format for the DRESP2 data is:

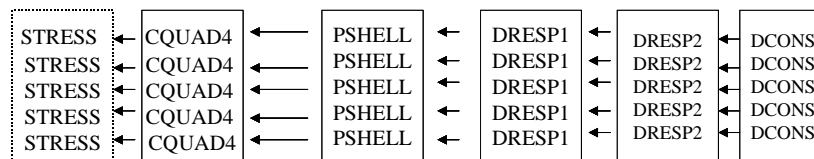
1	2	3	4	5	6	7	8	9	10
DRESP2	ID	LABEL	EQID	REGION					
+	"DVAR"	NDV1	NDV2	NDV3	NDV4	NDV5	NDV6	NDV7	NDV8
+		NDV9	-etc.-						
+	"DTABLE"	NC1	NC2	NC3	NC4	NC5	NC6	NC7	NC8
+		NC9	-etc.-						
+	"DGRID"	NG1	NGC1	NG2	NGC2	NG3	NGC3	NG4	NGC4
+		NG5	NGC5	-etc.-					
+	"DRESP1"	NR1	NR2	NR3	NR4	NR5	NR6	NR7	NR8
+		NR9	-etc.-						

This format allows to reference any DRESP1 data or any DVAR data. In case the DRESP1 or the DVAR data referenced are split by DSPLIT, it is important to understand what the program does.

Example: Consider the case where the user creates a DRESP2 that lists a DRESP1 that tags stress of a PSHELL that has 5 CQUAD4 elements:.



If the PSHELL data is split by DSPLIT, then program will internally create data equivalent to the following one:



If the DRESP2 references any split design variables, then the program will also split the DRESP2 in a similar way as described above. If the DRESP2 references multiple split design variables and/or multiple split DRESP1s, then all of them have to belong to the same topometry regions.

The above DRESP2 is termed a “split DRESP2”.

3.8.15 Using User-Supplied Subroutines with DRESP3 with Split DRESP1 and/or Split Design Variables

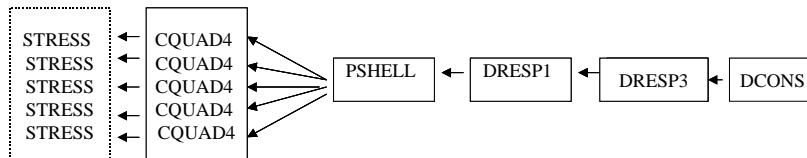
The format for the DRESP3 with user supplied subroutines is:

1	2	3	4	5	6	7	8	9	10
DRESP3	ID	LABEL	SUBID	REGION					
+	"DVAR"	NDV1	NDV2	NDV3	NDV4	NDV5	NDV6	NDV7	NDV8
+		NDV9	-etc.-						
+	"DTABLE"	NC1	NC2	NC3	NC4	NC5	NC6	NC7	NC8
+		NC9	-etc.-						
+	"DGRID"	NG1	NGC1	NG2	NGC2	NG3	NGC3	NG4	NGC4
+		NG5	NGC5	-etc.-					
+	"DRESP1"	NR1	NR2	NR3	NR4	NR5	NR6	NR7	NR8
+		NR9	-etc.-						

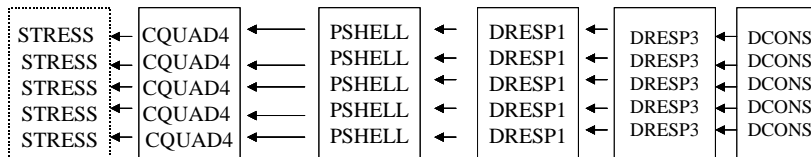
:

This format allows to reference any DRESP1 data or any DVAR data. In case the DRESP1 or the DVAR data referenced are split by DSPLIT, it is important to understand what the program does.

Example: Consider the case where the user creates a DRESP3 that lists a DRESP1 that tags stress of a PSHELL that has 5 CQUAD4 elements:



If the PSHELL data is split by DSPLIT, then program will internally create data equivalent to the following one:



If the DRESP3 references any split design variables, then the program will also split the DRESP3 in a similar way as described above. If the DRESP3 references multiple split design variables and/or multiple split DRESP1s, then all of them have to belong to the same topometry regions.

The above DRESP3 is termed a “split DRESP3”.

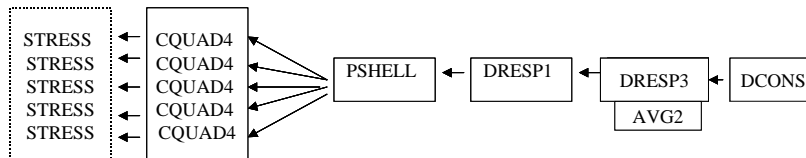
3.8.16 Using Built-in Equations in DRESP3 with Split DRESP1 and/or Split Design Variables

The format for the DRESP3 with built-in equations is:

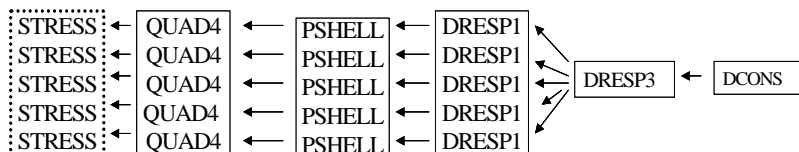
1	2	3	4	5	6	7	8	9	10
DRESP3	ID	LABEL	NAME	REGION					
+	"DVAR"	NDV1	NDV2	NDV3	NDV4	NDV5	NDV6	NDV7	NDV8
+		NDV9	-etc.-						
+	"DTABLE"	NC1	NC2	NC3	NC4	NC5	NC6	NC7	NC8
+		NC9	-etc.-						
+	"DGRID"	NG1	NGC1	NG2	NGC2	NG3	NGC3	NG4	NGC4
+		NG5	NGC5	-etc.-					
+	"DRESP1"	NR1	NR2	NR3	NR4	NR5	NR6	NR7	NR8
+		NR9	-etc.-						

This format allows to reference any DRESP1 data or any DVAR data. In case the DRESP1 or the DVAR data referenced are split by DSPLIT, it is important to understand what the program does.

Example: Consider the case where the user creates a DRESP3 that lists a DRESP1 that tags stress of a PSHELL that has 5 CQUAD4 element:.



In this case the program will not split the DRESP3 data, but rather will use the built-in equation with all the referenced responses.



In the above equation the result will be as one would expect:

$$R3 = ((\text{Stress } 1)**2 + (\text{Stress } 2)**2 + (\text{Stress } 3)**2 + (\text{Stress } 4)**2 + (\text{Stress } 5)**2)/5$$

3.8.17 Topometry Objective Function

Just like for any other form of optimization in the program any scalar DRESPx data can be selected as the objective function. Split DRESPx data can not be selected, as it does not have a scalar value.

3.8.18 Topometry Constraints

Any DRESPx data can be constrained (using DCONS), including split DRESPx data.

3.8.19 Practical Recommendations

In this section practical recommendations are given for topometry optimization

Do Not Select Element Responses in Large Topometry Problems

The reason is mostly for computational requirements. The program needs to compute sensitivities for all responses with respect to all design variables. If the problem has both a large number of design variables and a large number of responses, then the sensitivity matrix can grow very large.

3.8.20 Topometry Post Processing Result Files

Updated Input File

GENESIS can optionally print a file that contains all the updated properties as changed during the optimization process. The printing of this file is controlled with the **UPRINT** solution control command.

This file uses *GENESIS* input data format so that any preprocessor that can read *GENESIS* input files can be used to read this file. For more detail see **Updated Input File (pnameUPDATExx.dat)** (p. 831).

Sizing Post-Processing File

To facilitate visualizing topometry optimization, the property field values can be optionally printed in the post processing file named *pnameOPOSTxx.ext*, where *pname* is set to the base of the input filename, *xx* is the design cycle number, and *ext* is set according to the file format. The file format is set by the POST executive control command. The extension of the filename is based on the format: *op2* for OUTPUT2 format, *pch* for PUNCH format, *unv* for IDEAS format, *PST* for BINARY, FORMAT, PLOT or PATRAN format.

The Solution Control command, **SIZING** = POST, is used to generate this output.

The Solution Control command, **THICKNESS** = POST, is similar to **SIZING** = POST, except that it restricts output to just thickness of CQUAD4, CTRIA3 CQUAD8, CTRIA6 elements.

The property field values are printed using the element strain energy analysis result style.

For more detail see **Sizing Post-Processing Data (pnameOPOSTxx.ext)** (p. 845).

SSOL File

To facilitate visualizing shell sizing optimization results, shell elements can be converted to solid elements that reveal the thickness using the **SSOL** solution control command. The SSOL file is named *pnameSSOLxx.dat* and contains grid data and solid element connectivity (CHEXA and CPENTA) that reveal the thicknesses of CQUAD4 and/or CTRIA3 and/or CQUAD8 and/or CTRIA6 elements of the input data file.

The format of this file is the same as *GENESIS* input format. This file can be read directly by any preprocessor that can read *GENESIS* input data.

For more detail see **Shell to Solid Results File (pnameSSOLxx.dat)** (p. 847)

3.8.21 Topometry Data Example

Find the stiffest structure possible under one loading using 120 kg of the available material.

The topometry data for this problem could be:

	1	2	3	4	5	6	7	8	9	10
\$										
\$ Topometry designable regions										
\$										
DSPLIT	1	bottom		100						
DVPROP3	10	100	SOLID							
+	10									
DVAR	10	THICK	1.0	0.1	3.0					
\$ Mass constraints										
\$										
DRESP1	10	MASS	MASS							
DCONS	10			120.0						
\$										
\$ Objective function										
\$										
DRESP1	100	OBJ	SENERGY							
DOBJ	100	ENERGY								
\$										

In this example DSPLIT 1 is used to design all elements associated to property 100. Property 100 is designed used DVPROP3 10 and DVAR 10.

The DRESP1 10 is used to tag the system mass and the DCONS is used to keep the mass under 120.0kg.

The DRESP1 100 is used to tag the system strain energy and the DOBJ data is used to select the response 100 as the objective function of the problem.

In other words the problem is:

Minimize Strain Energy

Subject to;

$$\text{MASS} \leq 120\text{kg} \quad (\text{Eq. 3-65})$$

CHAPTER 4

Special Design Features

- Discrete Design Variables
- Random Variables for Reliability Analysis
- Design Variable Selection
- Equation Utility
- Adding to the Design Element Library
- Using External Analysis Programs
- Shifted Responses
- Restart Capability
- Mode Tracking
- Matching Measured Data
- Matching Eigenvector Components
- Relative Constraint Bounds
- Allowable Probability of Failure on Constraint Entries
- Mesh Smoothing
- Stress Ratio
- Customization Through Scripts

4.1 Discrete Design Variables

In certain classes of problems some elements cannot be designed using arbitrary numbers, that is, some design parameters can only take values from some finite set. For these classes of problems *GENESIS* provides data sets of discrete values. A set of discrete values can be input using the **DVSET** entry or the **DVSET1** entry.

The format of DVSET is:

1	2	3	4	5	6	7	8	9	10
DVSET	SID	VAL1	VAL2	VAL3	VAL4	VAL5	VAL6	VAL7	VAL8
+	VAL9	VAL10	etc.						

The SID is a nonunique set identification number. The VAL_i are the discrete values. For example:

1	2	3	4	5	6	7	8	9	10
DVSET	3	0.1	0.2	0.3	0.5	1.0	2.0	3.0	4.0
+	-0.1	-0.2							

The previous data will generate the following set:

Set = -0.2, -0.1, 0.1, 0.2, 0.3, 0.5, 1.0, 2.0, 3.0, 4.0

The DVSET1 data statement provides for a flexible way to automatically generate discrete values. The format of DVSET1 is:

1	2	3	4	5	6	7	8	9	10
DVSET1	SID	VAL1	VAL2	INCRE	NI				

As in DVSET, SID is a nonunique set identification number. VAL1 is the lowest value in the set, VAL2 is the highest discrete value in the set, INCRE is an increment value and NI correspond to the number of increments. Exactly two of VAL2, INCRE and NI must be non blank. The way *GENESIS* automatically generates the discrete values depends on which data are provided.

When VAL1, VAL2 and INCRE are provided, the discrete values defined by this data entry are given by $VAL_i = VAL1 + (i-1)INCRE$, $i = 1, 2, 3, \dots$ such that $VAL_i \leq VAL2$. VAL2 will be included in the set.

Example 1:

1	2	3	4	5	6	7	8	9	10
DVSET1	3	1.0	10.0	1.0					

The previous data will generate the following set:

Set = 1.0, 2.0, 3.0, 4.0, 5.0, 6.0, 7.0, 8.0, 9.0, 10.0

When VAL1, INCRE and NI are provided, the discrete values defined by this data entry are given by $VAL_i = VAL1 + (i-1)INCRE$ $i = 1, (NI + 1)$.

Special Design Features

Example 2:

1	2	3	4	5	6	7	8	9	10
DVSET1	3	1.0		1.0	9				

The previous data will generate the same set as in example 1:

Set = 1.0, 2.0, 3.0, 4.0, 5.0, 6.0, 7.0, 8.0, 9.0, 10.0

When VAL1, VAL2 and NI are provided, the discrete values defined by this data entry are given by $VAL_i = VAL1 + (i-1)(VAL2-VAL1)/NI$ $i = 1, (NI + 1)$.

Example 3:

1	2	3	4	5	6	7	8	9	10
DVSET1	3	1.0	10.0		9				

The previous data will generate the same set as in example 1 and 2:

Set = 1.0, 2.0, 3.0, 4.0, 5.0, 6.0, 7.0, 8.0, 9.0, 10.0

Multiple DVSET and DVSET1 entries may use the same set id. In this case, the set will be formed by the union of all corresponding entries.

4.1.1 Description of Use

Discrete sets are specified on the second field of the first continuation line on the DVAR entry. The basic format for the design variable entry that references a discrete set is:

1	2	3	4	5	6	7	8	9	10
DVAR	ID	LABEL	INIT	LB	UB				
+	DVSID								

Where DVSID is the set ID of discrete values. If the DVSID is INT, then *GENESIS* will use a set that contains integer numbers.

Discrete Optimization Starting Design Cycle and DOPT Parameter DSTART

GENESIS by default will perform first continuous optimization and then, after convergence, *GENESIS* will start performing discrete optimization. The user can change this option using the DOPT parameter DSTART. The options for DSTART are:

DSTART	Behavior
-2	<i>GENESIS</i> will ignore all discrete sets and perform only continuous optimization.
-1	<i>GENESIS</i> will start discrete optimization right after the continuous optimization process converges. (This is the default.)
0	<i>GENESIS</i> will perform discrete optimization starting with design cycle 0.
$N > 0$	<i>GENESIS</i> will perform discrete optimization after N continuous optimization design cycles. If <i>GENESIS</i> converges at the M^{th} design cycle, where $M < N$, then <i>GENESIS</i> will start the discrete optimization process in the $M+1$ design cycle.

Initial Design Variable Values and DOPT Parameter DVINIT2

If the initial value (INIT) given on the DVAR entry is not in the associated discrete set, then the behavior is determined by the DOPT parameter DVINIT2 as follows:

DVINIT2	Behavior
0	<i>GENESIS</i> will stop with fatal error. (This is the default.)
1	<i>GENESIS</i> will use the given initial value for the first analysis, and after that <i>GENESIS</i> will optimize using only the values in the discrete set.
2	<i>GENESIS</i> will replace the given initial value with the closest value in the set.
3	<i>GENESIS</i> will replace the given initial value with the closest value in the set. All discrete design variables will be held as constants during the optimization. This option allows restarts of a problem that has both discrete and continuous variables to keep trying to improve the design using only the continuous variables.

Bounds

The values in a discrete set that are outside the design variable bounds are not used. If there are no values in the discrete set within the bounds, then *GENESIS* will stop with a fatal error.

DLINK

Design variables that are selected as dependent design variables with the DLINK entry cannot reference discrete sets.

Example 1

User discrete set 12 = 0.1, 0.3, 0.5, 0.7

	1	2	3	4	5	6	7	8	9	10
DVSET	12	0.1	0.3	0.5	0.7					

Design variable 100; INIT = 0.49, LB= 0.3, UB = 0.55,

DVAR	100	THICK	0.49	0.3	0.55				
+	12								

Design variable 200; INIT = 0.49, LB= 0.24, UB = 0.75

DVAR	200	THICK	0.69	0.24	0.75				
+	12								

For design variable 100 *GENESIS* will search for the optimal value of the design variable in the following feasible set:

FEASIBLE DISCRETE SET = 0.3, 0.5

The discrete values 0.1 and 0.7 will not be used for design variable 100 because they are outside the bounds of this variable. *GENESIS* will use 0.3 as lower bound and 0.5 as upper bound for this design variable.

For design variable 200 *GENESIS* will search for the optimal value of the design variable in the following feasible set:

FEASIBLE DISCRETE SET = 0.3, 0.5, 0.7

The discrete value 0.1 will not be used for design variable 200 because it is outside the bounds of this variable. *GENESIS* will use 0.3 as lower bound and 0.7 as upper bound for this design variable.

Special Design Features

If the DVINIT2 parameter is not changed, *GENESIS* will stop with an error message because the initial design variables value 0.49 and 0.69 are not in their discrete sets. If the DOPT parameter DVINIT2 is changed to 1, then *GENESIS* will use the value 0.49 and 0.69 for the first analysis, but after that *GENESIS* will use values only in the corresponding feasible discrete sets. If DVINIT2 is set to 2, then *GENESIS* will use 0.5 as the initial value of design variable 100 and 0.7 as the initial value of design variable 200.

Example 2

User discrete set = 0.1, 0.3, 0.5, 0.7

1	2	3	4	5	6	7	8	9	10
DVSET	12	0.1	0.3	0.5	0.7				

Design variable 300; INIT = 0.49, LB= 0.35, UB = 0.495

DVAR	300	THICK	0.49	0.35	0.495				
+	12								

In this case, for design variable 300 there is not a feasible set of discrete values. *GENESIS* will stop giving an error message. To fix this data, the user would need to either to add some values into to the DVSET (e.g. 0.35 or 0.49 or 0.495) or to relax the bounds of the design variable (e.g. change LB to 0.3 or UB to 0.5).

4.2 Random Variables for Reliability Analysis

Reliability analysis is performed when one or more design variable are assigned a random distribution. A design variable is assigned a random distribution by providing a variability type and a variability value. These two inputs are specified in the **DVAR** data entry.

The format of the DVAR data entry with random inputs is shown next:

1	2	3	4	5	6	7	8	9	10
DVAR	ID	LABEL	INIT	LB	UB	DELX	DXMIN		
+	DVSID	VTYPE	VVALUE						

The variability type, VTYPE, and the variability value, VVALUE, are defined as follows:

- 3 VTYPE Variability type for random variable. One of "VAR", "STDDEV" or "COV" or blank. Default is STDDEV when VVALUE is non blank.
- 4 VVALUE Variability value for random variable (Real > 0 or blank).
 If VTYPE=VAR then value is variance.
 If VTYPE=STDDEV or blank then value is standard deviation.
 If VTYPE=COV then value is coefficient of variation

The variability types for random variables are defined as follows:

$$\text{STDDEV} = \text{Standard Deviation} = \sigma_X(X) = \sqrt{\text{Var}(X)}$$

$$\text{COV} = \text{Coefficient of Variation} = \text{COV}(X) = \frac{\sigma_X(X)}{\mu_X}$$

$$\text{VAR} = \text{Variance} = \text{Var}(X) = \frac{\sum_i^n (X_i - \mu_x)^2}{(n - 1)}$$

Example:

1	2	3	4	5	6	7	8	9	10
DVAR	100	Side	1.2	0.2	2.4				
+		STDDEV	0.1						

The previous data entry will create a random design variable that has a standard deviation of 0.1

Special Design Features

Reliability analysis, when performed, will be calculated for all retained constraints. The key values calculated are the reliability index and the probability of failure. Output for a stress constraint result is printed as follows:.

```
-----
ELEMENT STRESS  LOAD=      1  ELEMENT=      1  ITEM=      2  VALUE= 1.28536E+04
                                MEAN VALUE = 1.28536E+04
                                BOUND = 2.00000E+04
                                STANDARD DEVIATION = 1.74924E+03
                                BETA = 4.08543E+00
                                RELIABILITY = 9.99980E-01
                                PROBABILITY OF FAILURE = 2.00000E-05
```

Variables that are not assigned a random distribution, will be considered deterministic and will have a variance of 0.0. If no random variable exists, then traditional deterministic variables will be used

Example 2: Random variable with standard deviation of = 0.02

1	2	3	4	5	6	7	8	9	10
DVAR	20	AREA	2.0	1.0	5.0	0.5	0.05		
+		STDDEV	0.02						

The complete information that can be listed on a DVAR data entry with random inputs is shown next: :

Field Information Description

2	ID	Unique design variable identification number (Integer > 0).
3	LABEL	User supplied name for printing purposes (or blank).
4	INIT	Initial value. (Real). See remarks 5 and 6.
5	LB	Lower bound. (Real. $LB \leq INIT$).
6	UB	Upper bound. (Real. $UB \geq INIT$).
7	DELX	Design variable fractional move limit (Real > 0 or blank). See remark 1.
8	DXMIN	Design variable minimum move limit factor (Real > 0 or blank). See remark 1.
2	DVSID	Discrete design variable set identification number.(Integer > 0 or "INT" or blank). See remarks 2 and 3.
3	VTTYPE	Variability type for random variable. One of "VAR", "STDDEV" or "COV" or blank. Default is STDDEV when VVALUE is non blank. See Remarks 7 and 8.
4	VVALUE	Variability value for random variable (Real > 0 or blank). If VTTYPE=VAR then value is variance. If VTTYPE=STDDEV or blank then value is standard deviation. If VTTYPE=COV then value is coefficient of variation

4.3 Design Variable Selection

In certain classes of problems it is desirable to have a fraction of the number of design variable move to their upper bound while the rest to move to their lower bound. The **DSELECT** entry allows easy formulation of this type of “design variable selection” problem. The DSELECT entry lists the pool of design variables from which the subset to move to the upper bound is to be selected. The optimization process itself will find the optimal variables to move to their upper bound and the optimal variables to move to their lower bound. The optimization will be based on a user-given fraction along with an objective and, optionally, constraints. The objective and constraints could be from any *GENESIS* responses.

An example of the use of DSELECT is to find the optimal location of welds, where each weld is represented by CBUSH element designed by a design variable. In this case the objective could be to maximize the stiffness as measured by a torsion frequency. The constraints could be static displacements at selected grids. The fraction could be 0.2 to pick the best 20% of the candidate welds.

Another example would be the selection of one out of two candidate components. Each of the components could be represented by a superelement. Each superelement could be designed with a unique design variables that scale the stiffness of the superelement and take a values of 0 or 1. The fraction in this case would be 0.5. At the end of the optimization one design variable will have the value of 1.0 and the other 0.0. The superelement with a value of 1.0 will be the best of the two to keep.

4.3.1 Description of Use

Format

The format of DSELECT is

Format:

1	2	3	4	5	6	7	8	9	10
DSELECT	ID	LABEL	FRACT	BTYPE	GTYPE	TOL			
+	"DVAR"	NDV1	NDV2	NDV3	NDV4	NDV5	NDV6	NDV7	NDV8
+		NDV9	-etc.-						

The ID specifies a unique DSELECT identification number. The LABEL field is to provide a brief description or name. The field for FRACT is to specify the desired fraction. FRACT has to be between 0.0 and 1.0. BTYPE must be either UPPER or EXACT or blank (Default is EXACT). GTYPE is to specify the type of constraint *GENESIS* will generate to enforce the fraction. GTYPE could be AVG or ABS. TOL is a small number that represents a tolerance used by the optimizer.

Example 1: Create constraints so that 50% of the listed design variable move to their upper bound and 50% move to the lower bound.:

1	2	3	4	5	6	7	8	9	10
DSELECT	1	WeldFrac	0.5	EXACT	AVG				
+	DVAR	87	25	3	8				

Example 2: Create constraints so that the average sum of all listed normalized design variables do not exceed 50% of their upper bounds

1	2	3	4	5	6	7	8	9	10
DSELECT	1	WeldFrac	0.5	UPPER	AVG				
+	DVAR	87	25	3	8				

4.4 Equation Utility

The equation utility is one of the powerful features of *GENESIS*. It allows the designer the flexibility to create his/her own nonlinear relationships between design variables and properties of structural elements (i.e., area, thickness, bending stiffness, nonstructural mass, etc.) through the use of **DVPROP2** data. The equation utility also gives the designer the power to create his own responses which are nonlinear functions of structural responses (e.g., forces, stresses, displacements, frequencies, etc.), design variables, and grid locations through the use of **DRESP2** and **TRESP2** data.

This utility is easy to use, as the equation is specified in the input data in the standard Fortran format. A description of the use of the equation format and some specific examples which follow.

4.4.1 Description of Use

Format

The equation is specified on the **DEQATN** input data. The equation can actually be the result of a sequence of equations. Note that, as in Fortran, blanks are ignored. Each equation has a name that can be up to 31 alphanumeric characters. The names do not have to be distinct and are only for the user's reference. For the example below, an equation with an identification number of 20 has the name F1. Following the name are a list of arguments that are used in the equations. These arguments are enclosed in parentheses and must be from 1 to 31 alphanumeric characters beginning with a letter. The equation utility is case insensitive. This means that no distinction is made between upper and lower case letters. In fact the entire equation is transformed to upper case letters as it is read in.

1	2	3	4	5	6	7	8	9	10
DEQATN	20	F1(X1,X2)= X1 + X2							

Real constants can be placed in the equation using the exponential format (scientific notation). For example 140000., 140000, 140000.0, 1.4E+5, 14e4, 1.4E5.0, 1.4e5., 1400000E-1, 0.014e7, and .014E+7 are all evaluated as 140000.0.

Argument List

The argument list consists of design variables, structural responses, and nodal locations. Constants from the **DTABLE** input data may be used directly in equations, as long as the names are distinct from the function name and argument names. The constants are used to shorten the length of the equations and to reduce the amount of input data by defining a constant in one place that may be used in many equations. For example, if the constant PI is defined as 3.14159 on the DTABLE data, the equation

1	2	3	4	5	6	7	8	9	10
DEQATN	30	AREA(R)=3.14159*R**2							

could be replaced by.

1	2	3	4	5	6	7	8	9	10
DEQATN	30	AREA(R)=PI*R**2							

Note that the order that the arguments in the argument list does not have to coincide with the order that they appear in the equation. Also note that an argument can appear more than once in an equation. All the arguments in the argument list must appear at least once in the equation.

4

Layered Equations

The equation can actually be a sequence of equations separated by semi-colons (;). The arguments for all the equations are specified between parenthesis on the left-hand-side of the first equation. Arguments are not specified for the subsequent equations. The equations are evaluated in the order that they appear. The result of any preceding equation is used in a subsequent equation by specifying the preceding equation name as an argument. The value of the DEQATN is the value of the last equation. For example;

1	2	3	4	5	6	7	8	9	10
DEQATN	10	F1(A,B,C,D) = A+B ; F2 = A+D ; F3 = F1*F2*C							

In this example, F3 is calculated using the results of F1 and F2.

Order of Evaluation

The equations are evaluated from left to right in the following order:

1. Operations that are enclosed within parentheses.
2. Exponentiation.
3. Multiplication and division.
4. Addition and subtraction.
5. Relational tests greater than, greater or equal, less than, and less or equal
6. Relational tests equal and not equal

For example if $X1=3$ and $X2=6$ then

1	2	3	4	5	6	7	8	9	10
DEQATN	20	$F(X1,X2)=X2*X1/2 + X2/X1*2$							

gives a result of $9+4=13$.

Relational Operators

The relational operators are: $=$, \neq , $<$, \leq , $>$, \geq . These operators result in a value of 1.0 if the relation they are testing is true, or a value of 0.0 if the relation is false. These can be used to create piecewise functions with different formulas for different segments of the domain. For example:

1	2	3	4	5	6	7	8	9	10
DEQATN	20	$I(A) = (A<44)*(4.6248*A*A) + (A\geq 44)*(256.0*A-2300.0)$							

The relational operators should be used with caution. These operators can create nonsmooth and discontinuous functions that have discontinuous derivatives. The use of functions with discontinuous derivatives may slow the convergence of the optimization process.

Intrinsic Functions

The following intrinsic functions are supported by the equation utility: ABS, ACOS, ACOSH, ASIN, ASINH, ATAN, ATAN2, ATANH, ATANH2, AVG, COS, COSH, COTAN, DIM, EXP, INT, LOG, LOG10, LOGX, MAX, MIN, MOD, PI, RSS, SIGN, SIN, SINH, SQRT, SSQ, SUM, TAN, and TANH. Note that the input angle for the trigonometric functions is in radians and not degrees. The functions ABS, DIM, INT, MAX, MIN, MOD, and SIGN should be used with caution. These functions can create nonsmooth and discontinuous functions that have discontinuous derivatives. The use of functions with discontinuous derivatives may slow the convergence of the optimization process.

Integer Arithmetic

All numbers in the equation are treated as real numbers unless explicitly converted to integers using the INT intrinsic function. For example 14 is the same as 14.0. If X1=1 then the following function

1	2	3	4	5	6	7	8	9	10
DEQATN	25	F(X1)=INT(X1 + 5/2 + .6)							

is equal to INT(1.0 + 2.5 + 0.6) or INT(4.1) which equals 4.

Note that the INT function ignores the digits after the decimal point and does not round up. For example INT(5.7) is equal to 5. All exponentials and powers are treated as real numbers so that expressions like X2**4.5 and 1.3E2.6 are allowed. The only exception to this rule is when negative numbers are raised to a power. The expression -3**2 or -3**2.0 is treated as -3**INT(2). Note that the expression -3**2.5 will result in an error.

4.4.2 Use with DVPROP2 data

The **DVPROP2** data allows the designer to define element properties as nonlinear functions of the design variables. This is very important in the design of structures made up of BAR elements as the section properties are usually nonlinear functions of the cross sectional dimensions (design variables). For example take the design of a frame structure composed of constant thickness thin walled C-section BAR elements (see the figure below). Warping will be neglected. Suppose the thickness is 0.1 and the base (B) and height (H) are the design variables. The initial value for B will be 2 and for H will be 3. The design variables are allowed to be between 1.0 and 5.0.

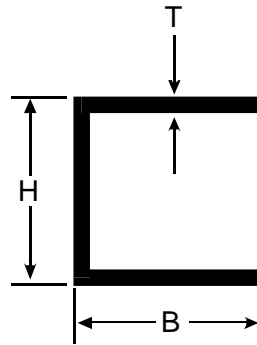


Figure 4-1C Section

The section properties in terms of the design variables are:

$$A = 2 \cdot B \cdot T + H \cdot T$$

$$I_Z = T \cdot H^3 / 12 + 2 \cdot (B \cdot T^3 / 12 + B \cdot T \cdot (H/2)^2)$$

$$I_Y = 2 \cdot (T \cdot B^3 / 12 + B \cdot T \cdot ((H \cdot T \cdot B / 2) / (2 \cdot B \cdot T + H \cdot T))^2) + H \cdot T^3 + H \cdot T \cdot (B/2 - (H \cdot T \cdot B / 2)) / (2 \cdot B \cdot T + H \cdot T)^2$$

$$J = (2 \cdot B + H) \cdot T^3 / 3.0$$

$$AS_Y = H \cdot T / 1.2$$

$$AS_Z = 2 \cdot B \cdot T / 1.2$$

The input data for this problem would be:

	1	2	3	4	5	6	7	8	9	10
DVAR		10	B	2.0	1.0	5.0				
DVAR		20	H	3.0	1.0	5.0				
\$										
DTABLE		T	0.1							
\$										
DEQATN	100	A(B,H)=2*B*T + H*T								
DEQATN	200	IZ(B,H)=T*H**3/12 + 2*(B*T**3/12 + B*T*(H/2)**2)								
DEQATN	300	IY(B,H)=2*(T*B**3/12 + B*T*((H*T*B/2)/(2*B*T + H*T))**2)								
+	+H*T**3+ H*T*(B/2-(H*T*B/2))/(2*B*T + H*T)**2									
DEQATN	400	J(B,H)=(2*B+H)*T**3/3.0								
DEQATN	500	ASY(H)=H*T/1.2								
DEQATN	600	ASZ(B)=2*B*T/1.2								
\$										
DVPROP2	1	22	2	100						
+	DVAR	10	20							
DVPROP2	2	22	4	200						
+	DVAR	10	20							
DVPROP2	3	22	5	300						
+	DVAR	10	20							
DVPROP2	4	22	3	400						
+	DVAR	10	20							
DVPROP2	5	22	7	500						
+	DVAR	20								
DVPROP2	6	22	8	600						
+	DVAR	10								

Note that:

1. The arguments are in the same order in the DVPROP2 data and the equation.
2. The specific section property is designated by the FID, which is the 4th value on the DVPROP2 data.

4.4.3 Use with DRESP2 and TRESP2 Data

The equation utility is used with the **DRESP2** and **TRESP2** capability to create nonlinear responses that are functions of design variable values, tabled constants, grid locations, and structural responses. Two example applications of the DRESP2 capability are the generation of “pass through” constraints in shape optimization and local column buckling constraints.

Pass Through Constraints

In the shape design of continuum structures, portions of the structure may pass through each other if they are not constrained from doing so (see **Figure 4-2**). In order to maintain a realistic design grid 91 must not pass through the flat surface defined by grids 40, 50, and 60. This is done by defining the surface and then constraining grid 91 to be above this surface. A surface is defined by three points using the equation:

$$\begin{aligned} & X*(Y1*(Z2-Z3)-Z1*(Y2-Y3)+(Y2*Z3-Z2*Y3)) - Y*(X1*(Z2-Z3) \\ & - Z1*(X2-X3)+(X2*Z3-Z2*X3)) + Z*(X1*(Y2-Y3)-Y1*(X2-X3) \\ & + (X2*Y3-Y2*X3)) - X1*(Y2*Z3-Z2*Y3)+Y1*(X2*Z3-Z2*X3) \\ & - Z1*(X2*Y3-Y2*X3) = 0 \end{aligned}$$

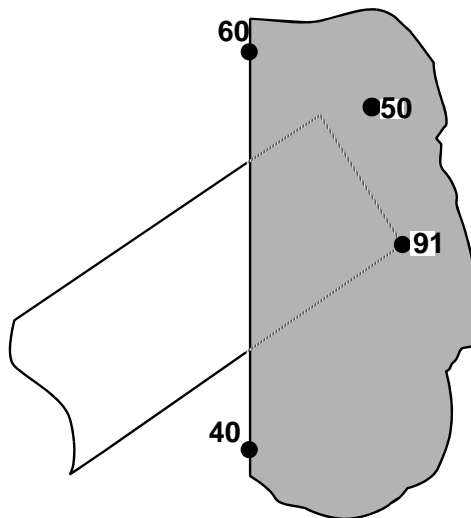


Figure 4-2Intersecting Parts

If the point (X,Y,Z) does not satisfy this equation then it is not on the plane. If the point (X,Y,Z) makes the right hand side of the equation greater than zero, then it is above the plane. Therefore, to keep point 91 from passing through the plane the value the right hand side of the equation must be greater than zero.

The constraint is then formulated as:

1	2	3	4	5	6	7	8	9	10
\$									
DCONS	10		0.0	1.0E30					
\$									
DRESP2	10	PASS	100						
+	DGRID	40	1	40	2	40	3	50	1
+		50	2	50	3	60	1	60	2
+		60	3	91	1	91	2	91	3
\$									
DEQATN	100	F(X1,Y1,Z1,X2,Y2,Z2,X3,Y3,Z3,X,Y,Z)=							
+	X*(Y1*(Z2-Z3)-Z1*(Y2-Y3)+(Y2*Z3-Z2*Y3)) -								
+	Y*(X1*(Z2-Z3)-Z1*(X2-X3)+(X2*Z3-Z2*X3)) +								
+	Z*(X1*(Y2-Y3)-Y1*(X2-X3)+(X2*Y3-Y2*X3)) -								
+	X1*(Y2*Z3-Z2*Y3)+Y1*(X2*Z3-Z2*X3)-Z1*(X2*Y3-Y2*X3)								

Note: Pass through constraints can be easily generated using DRESPG data.

Local Column Buckling Constraints

Consider a single round rod member in a 2D truss structure laying in the X-Z plane. The structure is being optimized using both sizing and shape design variables. Design variable 1 is the area of the truss member and design variables 31 and 32 control the X and Z location of grid 2. A DRESP2 response is constructed to prevent the rod member from buckling. Assume that a pin jointed rod made of a material with Young's Modulus of 1.0E7 will buckle if $FA*L^{**2}/A^{**2}$ is less than -0.7854E7 where FA is the axial force in the rod, L is its length, and A is its area. The length of the rod is equal to $SQRT((X1-X2)^{**2}+(Z1-Z2)^{**2})$ so the constraint equation is:

$$FA*((X1-X2)^{**2}+(Z1-Z2)^{**2})/A^{**2} \geq -0.7854E7 \quad (\text{Eq. 4-1})$$

Special Design Features

The important portion of the input data for this problem is shown below. Note that:

1. The DRESP2 is a nonlinear function of the shape (grid locations), sizing design variables (rod area), and structural response (force in the rod).
2. The order of the DVAR, DGRID, and DRESP1 on the DRESP2 data is fixed.

	1	2	3	4	5	6	7	8	9	10
\$										
GRID	1		1250.0	0.0	250.0		2456			
GRID	2		1000.0	0.0	250.0		2456			
CROD	1	1	1	2						
MAT1	1	1.0E7		0.1						
\$										
DVPROP 1	1	1	1							
+	1	1.0								
\$										
DVAR	1	A1	10.0	5.0	20.0					
DVAR	31	X3	0.0	-20.0	200.0		2.0			
DVAR	32	Z3	0.0	-200.0	200.0		2.0			
\$										
DVGRID	31	2		50.0	1.0					
DVGRID	32	2		50.0			1.0			
\$										
DRESP1	2001	2001	FORCE	ELEM		1	1			
\$										
DRESP2	201	B01	20							
+	DVAR	1								
+	DGRID	1	1	1	3	2	1	2	3	
+	DRESP1	2001								
\$										
DCONS	201	ALL	- 0.7854E7	1.0E30						
\$										
DEQATN	20	F1(A,X1,Z1,X2,Z2,FA)=FA*((X1-X2)**2+(Z1-Z2)**2) ; F = F1/A**2								

4

4.4.4 Error Messages

There are three types of error messages generated by the equation utility. The first type are format errors in the equation. These are quite straightforward and easy to correct. The second type is caused by having DVPROP2, DRESP2 or TRESP2 data that have a different number of arguments than the equations they reference. These first two types of errors are caught in the preprocessing stage of *GENESIS* and will cause the program to terminate with a fatal error.

The third type of error message is caused by overflows or invalid operations, such as ASIN(50.0), during the evaluation of the equations. These will cause the *GENESIS* to terminate. This may happen a far into the design process, so the user is cautioned to set up equations that are well conditioned. It is recommended to run small test problems before running large problems that contain suspect equations. In general, errors will occur if the equation causes the

1. Generation of numbers greater than 1.0E36 through multiplication or division,
2. EXP of a number larger than 83,
3. SIN, COS, or TAN of a number larger than 10.0E10, or
4. SINH or COSH of a number of magnitude greater than 83.

Some errors will not be caught by *GENESIS*, for example the addition of two large numbers causing overflow, and will lead to Fortran and system errors. Again, this will cause the program to terminate and valuable information may be lost. Below is a listing of the error messages generated by the equation utility. Please read them before using this powerful feature.

Error Messages Generated During the Preprocessing of the Equations

```
!!!!!! ERROR IN EQUATION 102 !!!!!!!
SOME FUNCTION NAME(S) CAN NOT BE FOUND IN THE
SET OF 'EQUATION' INTRINSIC FUNCTIONS
```

```
!!!!!! ERROR IN EQUATION 102 !!!!!!!
THE NUMBER OF THE LEFT PARENTHESES DOES NOT EQUAL THE NUMBER OF THE RIGHT
PARENTHESES
```

```
!!!!!! ERROR IN EQUATION 102 !!!!!!!
EQUATION HAS NO ARGUMENTS.
```

```
!!!!!! ERROR IN EQUATION 102 !!!!!!!
INVALID EQUATION.
```

```
!!!!!! ERROR IN EQUATION 102 !!!!!!!
ARGUMENT 12 IS NOT USED.
```

Special Design Features

*****! ERROR IN EQUATION 20 *****!
INCORRECT FORMAT OF INPUT FOR THE EQUATION. INVALID CHARACTER: "12X".
ARGUMENTS MUST BEGIN WITH A LETTER.

*****! ERROR IN EQUATION 22 *****!
INCORRECT FORMAT OF INPUT FOR THE EQUATION. TOO MANY DECIMAL POINTS IN A
FIELD. FIELD: 1.4E4.1.

*****! ERROR IN EQUATION 22 *****!
DOUBLE EXPONENT. FIELD: 1.4E4E3

*****! ERROR IN EQUATION 22 *****!
SUB-LEVEL EQUATION 2 IS NOT USED

*****! ERROR IN EQUATION 22 *****!
ARGUMENT IN SUB-LEVEL EQUATION OCCURS BEFORE IT IS DEFINED

*****! ERROR IN EQUATION 22 *****!
FIRST LEVEL EQUATION IS NOT USED

Error Messages Generated During the Preprocessing of the Design Input Data

ERROR ON DVPROP2 DATA ENTRY NUMBER 12. THE EQUATION ON THIS ENTRY REQUIRES 3 ARGUMENTS, BUT THIS DVPROP2 ENTRY HAS 4 ARGUMENTS.

ERROR ON DRESP2 DATA ENTRY NUMBER 3000. EQUATION NUMBER 102 REQUIRES 3 ARGUMENTS, BUT THIS DRESP2 ENTRY GENERATES 4 ARGUMENTS.

Error Messages Generated During the Evaluation of the Equations

!!! EQUATION ERROR: OVERFLOW DURING MULTIPLICATION !!!

!!! EQUATION ERROR: OVERFLOW DURING DIVISION !!!

!!! EQUATION ERROR: DIVIDE BY ZERO !!!

!!! EQUATION ERROR: RAISING NUMBER TO A POWER CAUSES OVERFLOW !!!

!!! EQUATION ERROR: NEGATIVE NUMBER RAISED TO A REAL POWER !!!

!!! EQUATION ERROR: SQUARE ROOT OF A NEGATIVE NUMBER !!!

!!! EQUATION ERROR: OVERFLOW DURING EXPONENTIAL OPERATION !!!

!!! EQUATION ERROR: LOG OF A NUMBER \leq ZERO !!!

!!! EQUATION ERROR: LOG10 OF A NUMBER \leq ZERO !!!

!!! EQUATION ERROR: SINE OF A LARGE NUMBER !!

!!! EQUATION ERROR: COSINE OF A LARGE NUMBER !!!

!!! EQUATION ERROR: TAN OF A LARGE NUMBER !!!

!!! EQUATION ERROR: COTAN OPERATION OVERFLOW !!!

!!! EQUATION ERROR: COTAN OF A LARGE NUMBER !!!

!!! EQUATION ERROR: ASIN OF A NUMBER > 1.0 OR < -1.0 !!!

!!! EQUATION ERROR: ACOS OF A NUMBER > 1.0 OR < -1.0 !!!

Special Design Features

!!! EQUATION ERROR: OVERFLOW DURING SINH OPERATION !!!

!!! EQUATION ERROR: OVERFLOW DURING COSH OPERATION !!!

The previous errors cause the following message to appear:

THE PREVIOUS ERROR OCCURRED DURING THE EVALUATION OF
EQUATION NUMBER 102. THE EQUATION ARGUMENTS ARE:
ARGUMENT # 1 = 30.00000
ARGUMENT # 2 = -6.230000

4.5 Adding to the Design Element Library

The *GENESIS* program has a design element library consisting of commonly used beam and plate element cross sections. These are used with the DVPROP3 data to formulate the relationship between the cross sectional dimensions and analysis model properties. The user may add up to 11 more cross sections to this library if the desired shapes are not present. This is accomplished by creating a shared object file (DLL) that has functions to calculate section properties from design variables and stress responses from design variables and forces. The **DLIB** executive control statement points to the shared object, and the **DLIB** bulk data statement makes the user-defined section available for **DVPROP3**.

4.5.1 Input Data for User Defined Design Elements (DLIB)

The **DLIB** bulk data entry is used to tell *GENESIS* the characteristics of the user's design element. The format of the DLIB data is:

1	2	3	4	5	6	7	8	9	10
DLIB	LIBID	TYPE	NCSD	NSP	NSPS	NSTR			
+	SP1	SP2	SP3	SP4	SP5	SP6			

The LIBID specifies the design element library number, and must be between 15 and 25. The design element TYPE must be either PBAR or PSHELL. The number of cross sectional dimensions used to define the cross section is NCSD. The number of section properties that will be calculated by the DLIBPROP interface function is NSP when shear deformation is excluded and by NSPS when shear deformation is included. The number of stresses calculated by the DLIBSTRESS interface function is NSTR. NSTR must be an even number (2 times the number of stress recovery points at each end) and be ≤ 16 . NSTR should be equal to zero if no stress recovery is needed. The SPi are the section property codes for the section properties calculated by the first user supplied subroutine. Note that there is a maximum of six possible section properties.

The section property codes are:

CODE	PROPERTY
1	Area
2	I2 (I_{yy})
3	I1 (I_{zz})
4	I - use if both I1 and I2 are the same
5	I12
6	J
7	Shear area AS_1 - Plane 1 (AS_y)
8	Shear area AS_2 - Plane 2 (AS_z)
9	Shear area - use if the same in both Planes 1 and 2
100	PSHELL plate thickness (T)
101	Plate bending stiffness (D)
102	Plate shear thickness (TS)

The section property codes must be listed in increasing numerical order. The shear deformation section properties, if they exist, should be listed last. If no shear deformation is to be included then NSP and NSPS should be the same. If there are less than six section properties then the last SPi should be left blank. Finally, note that, for BAR elements, if the I12 section property is calculated, then shear deformation must be excluded.

For example consider the constant thickness hat section in the figure below

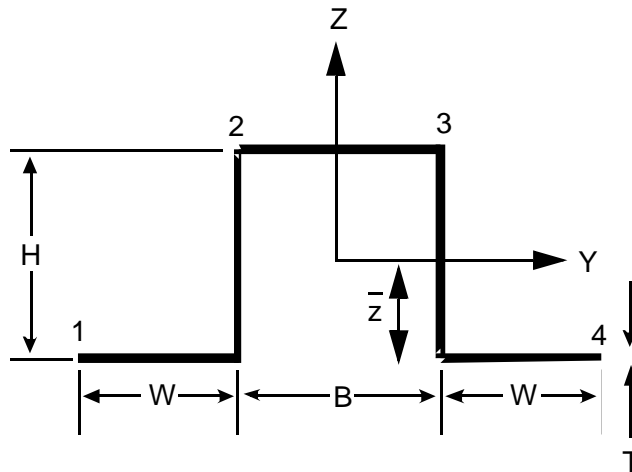


Figure 4-3 Hat Section

The section properties that will be calculated for this section are area, I1, I2, and the two shear areas. Four axial stresses will be calculated at each end at the points shown, for a total of eight stresses. This will be design element number 20. The input data for this element would be:

1	2	3	4	5	6	7	8	9	10
DLIB	20	PBAR	4	3	5	8			
+	1	2	3	7	8				

If shear deformation is not available the data would be:

1	2	3	4	5	6	7	8	9	10
DLIB	20	PBAR	4	3	3	8			
+	1	2	3						

4.5.2 User-Supplied Section Property Calculation Interface Function (DLIBPROP)

The user must write a subroutine that, given the element cross sectional dimensions, calculates the section properties. This function must be included in a shared object (DLL) that is specified by the **DLIB** executive control command. The Fortran name of the subroutine is DLIBPROP, and is declared as follows:

```
SUBROUTINE DLIBPROP(IWHICH, CSD, V, SP, IERR)
  INTEGER IWHICH
  DOUBLE PRECISION CSD(*)
  DOUBLE PRECISION V
  DOUBLE PRECISION SP(*)
  INTEGER IERR
```

IWHICH is input and is the library element number specified on the DLIB input data (15-25). This can be used to make a single shared object able to calculate multiple different user cross-sections. CSD is input and contains the design variable values identified by the DVPROP3. V is input and contains Poisson's ratio. SP is output and should contain the element properties identified by the DLIB bulk data. IERR is output and should contain 0 if no errors were detected. If an error is encountered in this routine the error switch IERR should be set to 1, causing the program to terminate. Possible errors include calculation of negative areas or principle bending moments of inertia. The DLIBPROP routine should set IERR = 2 if IWHICH corresponds to any value for which no user sections have been defined.

For the hat section the user-supplied subroutines might appear as:

```

SUBROUTINE DLIBPROP(IWHICH,CSD,V,SP,IERR)
  IMPLICIT NONE
C
  INTEGER IWHICH, IERR
C
  DOUBLE PRECISION CSD(*), V
C
  DOUBLE PRECISION SP(*)
C
  CALL MYPROP20(CSD, V, SP, IERR)

  RETURN
END
```


C

Special Design Features

```

C      .  SECTION PROPERTIES
      DOUBLE PRECISION ZBAR, AREA, IYY, IZZ, AS1, AS2

C
C      .  INITIALIZE ERROR CODE
      IERR = 0

C
C      .  GET CROSS SECTION DIMENSION PASSED FROM CALLER
C      .  NOTE: THE ORDER OF THE SECTION DIMENSIONS IS SPECIFIED ON THE
C      .      DVPROP3 BULK DATA ENTRY FOR THIS USER CROSS SECTION
C      .      (ELTYPE = 20)
      T = CSD(1)
      H = CSD(2)
      W = CSD(3)
      B = CSD(4)

C
C      .  COMPUTE SECTION PROPERTIES
      AREA = T*(B+2.0D0*(W+H))

C
      ZBAR = (2.0D0 * H*T*(0.5D0*H) + B*T*H)/AREA

C
      IYY = B*T*T*T/12.0D0 + B*T * (H-ZBAR)**2 +
$      2.0D0 * (T*H*H*H/12.0D0 + T*H * (0.5D0*H-ZBAR)**2 ) +
$      2.0D0 * (W*T*T*T/12.0D0 + T*W * ZBAR**2 )

C
      IZZ = T*B*B*B/12.0D0 +
$      2.0D0 * ( H*T*T*T/12.0D0 + T*H * (0.5D0*B)**2 ) +
$      2.0D0 * ( T*W*W*W/12.0D0 + T*W * (0.5D0*(B+W))**2 )

C
      AS1 = T*B + 2.0D0 * T*W

C
      AS2 = 2.0D0 * T*H

C
C      .  CHECK FOR PHYSICALLY UNREALIZABLE PROPERTIES
      IF(AREA.LE.0.0D0 .OR. IYY.LE.0.0D0 .OR. IZZ.LE.0.0D0 .OR.
$      AS1.LE.0.0D0 .OR. AS2.LE.0.0D0) IERR = 1

C
C      .  PASS SECTION PROPERTIES BACK TO CALLER
C      .  NOTE: THE ORDER OF SECTION PROPERTIES IS SPECIFIED ON THE
C      .      DLIB BULK DATA ENTRY FOR THIS USER CROSS SECTION (LIBID = 20)
      SP(1) = AREA
      SP(2) = IYY
      SP(3) = IZZ
      SP(4) = AS1
      SP(5) = AS2

C
      RETURN
      END

```

Note that the section properties must be in the same order as specified on the DLIB data. The order of the cross sectional dimensions in CSD is determined by the order that they are listed in the DVPROP3 input data.

If a language other than Fortran is used to create the shared object, care must be taken to ensure that the interface function has the correct name and arguments. The actual required function names are system dependent. For example, using the C language, the DLIBPROP interface function should have the following prototype:

Microsoft Windows	<code>__declspec(dllexport) void DLIBPROP(int *iwhich, double csd[], double *v, double sp[], int *ierr);</code>
Solaris, Linux, HP-UX	<code>void dlibprop_(int *iwhich, double csd[], double *v, double sp[], int *ierr);</code>
AIX	<code>void dlibprop(int *iwhich, double csd[], double *v, double sp[], int *ierr);</code>

4.5.3 User-Supplied Stress Recovery Calculation Interface Function (DLIB STRESS)

If the user wishes to calculate stresses in the user defined design element, then a second user written interface must be exported by the shared object. The Fortran name of the subroutine is DLIBSTRESS, and is declared as follows:

```
SUBROUTINE DLIBSTRESS(IWHICH, CSD, FORCE, STRESS, IERR)
  INTEGER IWHICH
  DOUBLE PRECISION CSD(*)
  DOUBLE PRECISION FORCE(*)
  DOUBLE PRECISION STRESS(*)
  INTEGER IERR
```

IWHICH is input and is the library element number specified on the DLIB input data (15-25). This can be used to make a single shared object able to calculate multiple different user cross-sections. CSD is input and contains the design variable values identified by the DVPROP3. FORCE is input and contains the element forces. STRESS is output and should contain the element stresses desired. IERR is output and should contain 0 if no errors were detected. If an error is encountered in this routine the error switch IERR should be set to 1, causing the program to terminate. The DLIBSTRESS routine should set IERR = 2 if IWHICH corresponds to any value for which no user sections have been defined.

For the hat section the user-supplied subroutines might appear as:

```
SUBROUTINE DLIBSTRESS(IWHICH,CSD,FORCE,STRESS,IERR)
  IMPLICIT NONE

C
  INTEGER IWHICH, IERR
C
  DOUBLE PRECISION CSD(*), FORCE(*)
C
  DOUBLE PRECISION STRESS(*)
C
  CALL MYSTRESS20(CSD, FORCE, STRESS, IERR)

  RETURN
  END

C
C . USER ROUTINE TO RECOVER STRESS FOR POINTS ON A HAT SHAPED
C . BAR SECTION
C . NOTE: THE NAME OF THIS ROUTINE INDICATES THAT IT IS FOR
C . USER CROSS SECTION (LIBID) = 20
C
C . ARGUMENTS
C . CSD(*), DOUBLE PRECISION, IN, CROSS SECTIONAL DIMENSIONS
C . FORCE(*), DOUBLE PRECISION, IN, BEAM FORCES
C . STRESS(*), DOUBLE PRECISION, OUT, STRESS RESPONSES
C . IERR, INTEGER, OUT, ERROR CODE
C
```

```

SUBROUTINE MYSTRESS20(CSD,FORCE,STRESS,IERR)
C  IMPLICIT NONE
    DOUBLE PRECISION CSD(*)
    DOUBLE PRECISION FORCE(*)
    DOUBLE PRECISION STRESS(*)
    INTEGER IERR
C
C  .  HAT SECTION DIMENSIONS
    DOUBLE PRECISION T,H,W,B
C
C  .  DUMMY VARIABLE REQUIRED FOR CALL TO GNLB20
    DOUBLE PRECISION V
C
C  .  BEAM SECTION PROPERTIES
    DOUBLE PRECISION SP(5), AREA, IYY, IZZ, ZBAR
C
C  .  BEAM ELEMENT FORCES
    DOUBLE PRECISION FA, FA1, FA2, TA, MA1, MA2,
$      FB, FB1, FB2, TB, MB1, MB2
C
C  .  INTERMEDIATE VARIABLES FOR STRESS CALCULATIONS
    DOUBLE PRECISION SA1I,SA1O,SA2T,SA2B,SA3,
$      SB1I,SB1O,SB2T,SB2B,SB3
C
C  .  INITIALIZE ERROR CODE
    IERR = 0
C
C  .  GET BEAM FORCES FROM CALLER
C  .  NOTE: THE MEANING OF THE FORCES IN THE FORCE ARRAY IS THE SAME
C  .  AS IS SPECIFIED BY THE DRESP1 FORCE ITEM CODE FOR A BAR
    FA = FORCE(1)
    FA1 = FORCE(2)
    FA2 = FORCE(3)
    TA = FORCE(4)
    MA2 = FORCE(5)
    MA1 = FORCE(6)
    FB = FORCE(7)
    FB1 = FORCE(8)
    FB2 = FORCE(9)
    TB = FORCE(10)
    MB2 = FORCE(11)
    MB1 = FORCE(12)
C
C  .  GET CROSS SECTION DIMENSION PASSED FROM CALLER
C  .  NOTE: THE ORDER OF THE SECTION DIMENSIONS IS SPECIFIED ON THE
C  .  DVPROP3 BULK DATA ENTRY FOR THIS USER CROSS SECTION
C  .  (ELTYPE = 20)
    T = CSD(1)
    H = CSD(2)
    W = CSD(3)
    B = CSD(4)

```

Special Design Features

```

C
C      .   GET SECTION PROPERTIES
V = 0.3D0
CALL MYPROP20(CSD,V,SP,IERR)
IF(IERR.NE.0) RETURN

C
      AREA = SP(1)
      IYY = SP(2)
      IZZ = SP(3)

C
      ZBAR = (2.0D0 * H*T*(0.5D0*H) + B*T*H)/AREA

C
C      .   STRESSES DUE TO AXIAL FORCE
SA3 = FA/AREA
SB3 = FB/AREA

C
C      .   STRESSES DUE TO BENDING
SA1I = 0.5D0*B * MA1/IZZ
SB1I = 0.5D0*B * MB1/IZZ
SA1O = (0.5D0*B+W) * MA1/IZZ
SB1O = (0.5D0*B+W) * MB1/IZZ
SA2T = (H-ZBAR) * MA2/IYY
SB2T = (H-ZBAR) * MB2/IYY
SA2B = ZBAR * MA2/IYY
SB2B = ZBAR * MB2/IYY

C
C      .   PASS STRESSES BACK TO CALLER
C      .   NOTE:  THE NUMBER OF STRESS RESPONSES MUST EQUAL WHAT WAS
C      .             SPECIFIED ON THE DLIB CARD FOR THIS USER CROSS SECTION
C      .             (LIBID = 20).  THE ORDER OF THESE STRESS RESPONSES AT
C      .             THE DISCRETION OF THE USER.  THE STRESS ITEM CODE OF A
C      .             DRESP1 ENTRY FOR THIS BEAM ELEMENT WILL SELECT THE
C      .             CORRESPONDING ENTRY OF THIS STRESS ARRAY.
STRESS(1) = SA1O + SA2B + SA3
STRESS(2) = SA1I - SA2T + SA3
STRESS(3) = - SA1I - SA2T + SA3
STRESS(4) = - SA1O + SA2B + SA3
STRESS(5) = SB1O + SB2B + SB3
STRESS(6) = SB1I - SB2T + SB3
STRESS(7) = - SB1I - SB2T + SB3
STRESS(8) = - SB1O + SB2B + SB3

C
      RETURN
      END

```

Note that the section properties are calculated by calling MYPROP20. Because of this the vector SP(5) must be defined. The forces are defined in the order shown.

If a language other than Fortran is used to create the shared object, care must be taken to ensure that the interface function has the correct name and arguments. The actual required function names are system dependent. For example, using the C language, the DLIBSTRESS interface function should have the following prototype:

Microsoft Windows	<code>__declspec(dllexport) void DLIBSTRESS(int *iwhich, double csd[], double force[], double stress[], int *ierr);</code>
Solaris, Linux, HP-UX	<code>void dlibstress_(int *iwhich, double csd[], double force[], double stress[], int *ierr);</code>
AIX	<code>void dlibstress(int *iwhich, double csd[], double force[], double stress[], int *ierr);</code>

4.6 Using External Analysis Programs

In this section, the coupling of an external program with *GENESIS* is described. This feature is useful to include information in the design process which is not otherwise defined by *GENESIS*.

The *GENESIS* program contains interfaces that allow the user to link their own analysis programs to it. Although *GENESIS* is written in Fortran, the user's programs can be written in any language. The output response values from the users program can be constrained or used as the objective function in the optimization portion of *GENESIS*. This capability allows the user to use special or proprietary analysis programs in conjunction with the standard structural analysis capability present in *GENESIS*. For example, this feature can be useful for the generation of an objective function that accounts for the cost of a structure.

4.6.1 Capabilities

There are two different methods available to interface a user's analysis program to *GENESIS*. The first uses the DRESPU bulk data and the GNUSER external program. The second uses the DRESP3 or TREPS3 bulk data and the DRESP3 interface function.

DRESPU

The DRESPU method works well when a large external analysis capability must be used with *GENESIS*. Since the user program is called only once per design cycle, or one plus the number of design variables per design cycle for finite difference gradients, the user program can be quite CPU time intensive. The disadvantage of this method is that only the design variable values can be used as input data. No *GENESIS* calculated responses or finite element mesh data can be sent to the user program

GENESIS is set up to call an external program to evaluate the user responses. When using this option, user-written code does not need to be linked directly with *GENESIS*, and therefore, the user can use any programming language to write the GNUSER program.

When *GENESIS* needs to evaluate the user responses, it will call the executable program specified by the **GNUSER** executive control command. Three arguments will be sent as command-line options to the user program. The first argument is the name of a file, written by *GENESIS*, that contains the design variable values, one value per line (sorted in ascending order of design variable ID). The second argument is the name of a file that should be created by the user program and should contain as many response values, one per line, as there are DRESPU data statements in the input data file. The data should be sorted in ascending DRESPU ID order. The third argument will be an integer. "0" indicates that responses should be calculated.

The user program only needs to evaluate responses, not sensitivities, as *GENESIS* can use finite difference to calculate any gradients that it needs. If desired, the user program can also calculate gradients. In this case, set the DOPT parameter **IUGRAD** to 1. When *GENESIS* runs the user program for gradients, the third command line argument will be an integer > 0. This value is the number of responses for which the gradient should be calculated. Additionally, there will be a fourth command line argument. The fourth argument is the name of a file, written by *GENESIS*, that contains the response indices, one number per line. These indices identify the responses for which the gradient should be calculated. Note that these are response indices, and not DRESPU ID values. When gradients are requested by *GENESIS*, the second command line argument is the name of a file that should be created by the user program to communicate the gradient values back to *GENESIS*. In this case, response values are not needed, and only gradients should be written to the file, one number per line. The first NDVAR lines should contain the gradient of the first response index with respect to each of the design variables, where NDVAR is the number of design variables. The next NDVAR lines correspond to the second response index, and so on.

DRESP3 / TRESP3

Subroutines are available for user responses. The input data from *GENESIS* to the DRESP3 interface function is defined in the same manner as the DRESP2 or TRESP2 data. Design variable values, table constants, grid point locations, and DRESP1/DRESPG responses calculated by *GENESIS* can be used as input data to DRESP3. Extra variable values, table constants and TRESP1 responses calculated by *GENESIS* can be used as input data to TRESP3. The output from these subroutines are used as constraints and/or the objective function response. The sensitivities of the DRESP3 / TRESP3 responses are calculated automatically by *GENESIS* using the finite difference method. The DRESP3 interface function is called many times per design cycle. For this reason, these subroutines should not be CPU intensive.

Which method to use?**4**

For user programs that are very CPU intensive, the DRESPU capability must be used. The disadvantage of the DRESPU capability is that only the design variable values can be used as input data to the user program. For user responses that require structural responses calculated by *GENESIS* or grid point locations, the DRESP3 capability must be used. The disadvantage of the DRESP3 capability is that the user program is called many times during the optimization process. For this reason the user program cannot be CPU intensive.

For topology optimization problems, there is no choice but to use the TRESP3 capability. The user program is called many times during the optimization process. For this reason the user program cannot be CPU intensive.

Use of the DRESPU capability is explained in Sections [4.6.2](#) through [4.6.3](#).

Use of the DRESP3 / TRESP3 capability is explained in Sections [4.6.4](#) through [4.6.7](#).

4.6.2 Use of the DRESPU Capability

In this section, the coupling of an external program with *GENESIS* is described. This feature is useful to include information in the design process which is not otherwise defined by *GENESIS*.

The DRESPU input data

The use of the user analysis program responses in the formulation of the optimization problem is controlled by the **DRESPU** input data. The format of this data is:

1	2	3	4	5	6	7	8	9	10
DRESPU	ID	LABEL	LB	UB					

The ID corresponds to the response ID and must be unique with respect to other DRESPU, DRESP1, DRESP2, DRESP3 and DRESPG ID's. The LABEL is used to identify the response in printed output and is optional. LB and UB are the lower and upper bounds on the response and are required, even for the objective function response. The lower and upper bounds generate two constraints per response. These constraints are screened by only retaining those whose values are larger than the constraint truncation threshold for the DRESP2 (equation) responses during each design cycle. If this response is the objective function, set LB = -1.0e+30 and UB = 1.0e+30.

4

User program response as the objective function

If a DRESPU response corresponds to the objective function then it must be referenced by the DOBJ input data. Note that the LID data on the DOBJ data is ignored in this case. The LB and UB data on the DRESPU data entry are still required and still generate constraints so use large values for these data.

Gradient calculations

The procedure for obtaining the DRESPU response gradients is controlled by the IUGRAD value on the DOPT input data. If IUGRAD=0, the default, *GENESIS* will automatically calculate the gradients by finite difference, making repeated calls to the users analysis program. If IUGRAD=1, then the user's analysis program must supply the gradients.

4.6.3 GNUSER: The GENESIS External User Program

Call Interface

GENESIS will run the external program as if the program were started from a command line with three or four command line arguments:.

```
user_program dvar_file return_file 0
```

or

```
user_program dvar_file return_file nsens sens_index_file
```

The three argument version will be used whenever *GENESIS* wants the responses calculated. If the input data specifies that the user program will calculate gradients, then when *GENESIS* wants the sensitivities calculated, the four argument version will be used. In this case, the third argument, *nsens*, will be an integer > 0 .

user_program is the name of the external user executable as specified on the **GNUSER** executive control command.

dvar_file is the name of a file created by *GENESIS*. This file will contain the design variable values, one number per line (sorted in ascending order of design variable ID). There will be *NDVAR* values in this file, where *NDVAR* is the number of design variables defined by the input data. Note that this can be greater than the number of **DVAR** entries if **DSPLIT** or **DTGRID** are used.

return_file is the name of a file that should be created by the user program containing either response values or gradient values. If responses are to be calculated, then they should be written to this file, one number per line (sorted in ascending order of DRESPU ID). If gradients are to be calculated, then they should be written to this file, one number per line. The first *NDVAR* values should be the gradient of the first response index with respect to each of the design variables. The next *NDVAR* values should be the gradient of the second response index, and so on. There should be a total of $NDVAR * nsens$ gradient values.

nsens is the number of retained responses (i.e., the number of responses that need gradients calculated). This can also be used as a switch to determine what the user program should do. If the third argument is 0, then the program should calculate responses. If the third argument is > 0 , then the program should calculate gradients. If the DOPT parameter **IUGRAD** is set to 0 (the default), then the third argument will always be 0, and the user program will only need to calculate responses.

sens_index_file is the name of a file created by *GENESIS*. This file will contain the response indices, one number per line. Each response index identifies a response for which the gradient is desired. Note that these are not DRESPU ID values, but rather indices to the position of the response as returned in the return file.

Gradient Calculations

If the user's program is able to calculate the gradients of the responses with respect to the design variables, then **IUGRAD** should be set equal to 1 on the DOPT input data. *GENESIS* will first call GNUSER with *nsens*=0 to get the response values. After the constraints on these responses have been evaluated and screened, *GENESIS* will call GNUSER again to get the gradients of the retained responses. In this case *nsens* will be the number of responses for which the gradient is needed. The *sens_index_file* contains a list of the retained responses. For example if there are five user responses and numbers 1, 4, and 5 are retained, *sens_index_file* will contain {1,4,5}. For this example, the gradient of response 1 would be written first to the *return_file*, the gradient of response 4 would be written second, and finally, the gradient of response 5 written third. Note that the response index corresponds to the position of the response in the response vector and not the DRESPU ID. For example if the DRESPU ID's are 5, 10, 2, 7, and 100 then the second response corresponds to the DRESPU data with ID=5 (because the responses will be sorted by increasing DRESPU ID). Also note that each gradient is in the order of increasing design variable ID. It is recommended, although not required, to number both the design variables and DRESPU data in order starting with 1 and not skipping any numbers. This may lead to fewer errors.

To calculate the gradients using finite difference (*IUGRAD* = 0), *GENESIS* uses two DOPT parameters; **FDCHU** and **FDCHMU**.

GNUSER Example

The following example user program is setup to calculate both responses and gradients.

```

PROGRAM GNUSER
    IMPLICIT NONE

    C
    C      ! THIS PROGRAM SHOULD KNOW HOW MANY DESIGN VARIABLES
    C      ! THERE ARE.  HOWEVER, SOME RESPONSES ARE FLEXIBLE ENOUGH
    C      ! TO NOT DEPEND ON THE NUMBER (E.G., SUM ALL THE DVARS)
    C
    C      ! MAXDV IS THEREFORE EITHER THE EXACT NUMBER OF DVARS
    C      ! OR AN UPPER BOUND ON THE NUMBER
    C
    C      ! WARNING -- IF THERE ARE MORE THAN MAXDV VALUES IN THE FILE
    C      ! THIS PROGRAM WILL SILENTLY IGNORE THE EXTRA ONES
    C
    INTEGER MAXDV
    PARAMETER (MAXDV = 5000)
    DOUBLE PRECISION DVARS(MAXDV)

    C
    C      ! THIS PROGRAM MUST KNOW HOW MANY RESPONSES IT CALCULATES
    C
    C      ! SET NRESPU TO THE NUMBER OF RESPONSES
    C
    C      ! WARNING -- IF IT CALCULATES MORE THAN GENESIS EXPECTS,
    C      ! THEN GENESIS WILL SILENTLY IGNORE THE EXTRA RESPONSES

```

Special Design Features

```

C      ! (GENESIS EXPECTS ONE RESPONSE PER DRESPU ENTRY IN THE
C      !   INPUT FILE)
C
      INTEGER NRESPU
      PARAMETER (NRESPU = 2)
      DOUBLE PRECISION RESPU(NRESPU)
      INTEGER ICUSER(NRESPU)
      DOUBLE PRECISION DAVE(MAXDV)
      DOUBLE PRECISION DSDEV(MAXDV)

C
C      !   LOCAL VARIABLE DECLARATIONS
C
      INTEGER NARG
      CHARACTER*80 DVFILE, RSFILE
      CHARACTER*8  CICODE
      CHARACTER*80 ICFILE
      DOUBLE PRECISION VAL

C
      INTEGER NTM, NDV, I, J, IR, ICODE, IERR
      LOGICAL LEX

C
      INTEGER IARGC
      EXTERNAL IARGC

C
C      !   SET UNIT NUMBERS FOR ERROR MESSAGES
C
      NTM = 0

C
C      !   GET THE DESIGN VARIABLE AND RESPONSE FILE NAMES
C
      NARG = IARGC()
      IF (NARG.LT.2) THEN
        WRITE (NTM,1998)
1998      FORMAT (
$5X,'AN ERROR OCCURRED IN THE USER-RESPONSE PROGRAM:'/
$5X,'TWO COMMAND LINE ARGUMENTS EXPECTED'//)
        GO TO 9999
      ENDIF
      CALL GETARG(1,DVFILE)
      CALL GETARG(2,RSFILE)
      IF (NARG.GE.4) THEN
        CALL GETARG(3,CICODE)
        READ (CICODE, '(I8)') ICODE
        IF (ICODE .GT. 0) THEN
          CALL GETARG(4,ICFILE)
          INQUIRE (FILE=ICFILE, EXIST=LEX, IOSTAT=IERR, ERR=999)
          IF (.NOT.LEX) THEN
            WRITE (NTM,2997)
2997      FORMAT (
$          5X,'AN ERROR OCCURRED IN THE USER-RESPONSE PROGRAM:'/
$          5X,'ICUSER FILE NOT FOUND'//)

```

```

        GO TO 9999
    ENDIF
C
    OPEN (12, FILE=ICFILE, STATUS='OLD', FORM='FORMATTED',
*       IOSTAT=IERR, ERR=999)
    DO I=1,ICODE
        READ(12,'(I8)',IOSTAT=IERR,ERR=999) ICUSER(I)
    END DO
    CLOSE(12)
    ENDIF
    ELSE
        ICODE = 0
    ENDIF
C
C    ! READ THE DESIGN VARIABLE VALUES FROM THE DV FILE
C
    INQUIRE (FILE=DVFILE, EXIST=LEX, IOSTAT=IERR, ERR=999)
    IF (.NOT.LEX) THEN
        WRITE (NTM,2998)
2998    FORMAT (
        $5X,'AN ERROR OCCURRED IN THE USER-RESPONSE PROGRAM:'/
        $5X,'DESIGN VARIABLE FILE NOT FOUND'/)
        GO TO 9999
    ENDIF
C
    OPEN (12, FILE=DVFILE, STATUS='OLD', FORM='FORMATTED',
*       IOSTAT=IERR, ERR=999)
C
    NDV = 0
    DO I=1,MAXDV
        READ(12,'(F20.0)',IOSTAT=IERR,ERR=999,END=3000) VAL
        NDV = NDV + 1
        DVAR$ (NDV) = VAL
    END DO
3000 CONTINUE
C
    CLOSE(12)
C
    OPEN (12, FILE=RSFILE, STATUS='UNKNOWN', FORM='FORMATTED',
$       IOSTAT=IERR, ERR=999)
C
    IF (ICODE .EQ. 0) THEN
C        ! CALCULATE RESPONSES HERE
C
C        ! COMPUTE AVERAGE AND STANDARD DEVIATION
C
        CALL AVESTA(DVAR$,NDV,RESPU(1),RESPU(2))
C
C        ! WRITE THE RESPONSE VALUES IN THE RESPONSE FILE
C
        DO I=1,NRESPU

```

Special Design Features

```

        WRITE(12,'(1P,E20.13)') RESPU(I)
    END DO
ELSE
C      !  CALCULATE SENSITIVITIES HERE
    CALL DAVESTA(DVARS,NDV,DAVE,DSDEV)
    DO I=1,ICODE
        IR = ICUSER(I)
        IF (IR .EQ. 1) THEN
            DO J=1,NDV
                WRITE(12,'(1P,E20.13)') DAVE(J)
            END DO
        ELSE IF (IR .EQ. 2) THEN
            DO J=1,NDV
                WRITE(12,'(1P,E20.13)') DSDEV(J)
            END DO
        ELSE
            DO J=1,NDV
                WRITE(12,'(1P,E20.13)') 0.0D0
            END DO
        ENDIF
    END DO

    ENDIF

C
    CLOSE (12)
C
C      !  FINISHED
C
    GO TO 9999
C
    999  CONTINUE
        WRITE (NTM,9998) IERR
    9998 FORMAT (
        $5X,'AN I/O ERROR OCCURRED IN THE USER-RESPONSE PROGRAM:'/
        $5X,'IOSTAT = ',I8/)
C
    9999 CONTINUE
        END
C
C*****
C      ***
C
C      PROGRAM NAME: AVESTA
C
C      DVAR(NDVAR) : DESIGN VARIABLE VALUES
C
C      NDVAR : NUMBER OF DESIGN VARIABLE
C      NRESP : NUMBER OF RESPONSE
C
C*****
C      ***

```



```

C
      SUBROUTINE AVESTA(DVAR,NDVAR,AVE,SDEV)
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      DOUBLE PRECISION DVAR(NDVAR)

C
C      ! COMPUTE AVERAGE
C
      AVE = 0.0D0

C
      DO I = 1, NDVAR
        AVE = AVE + DVAR(I)
      END DO

C
      AVE = AVE/REAL(NDVAR)

C
C      ! COMPUTE VARIATION
C
      VAR = 0.0D0

C
      DO I = 1, NDVAR
        VAR = VAR + (AVE - DVAR(I))**2/REAL(NDVAR)
      END DO

C
C      ! COMPUTE STANDARD DEVIATION
C
      SDEV = DSQRT(VAR)

C
      RETURN
      END
C*****
C      ***
C
C      PROGRAM NAME: DAVESTA
C
C      PURPOSE: COMPUTE DERIVATIVES OF AVERAGE AND STANDARD DEVIATION.
C
C      NDVAR : NUMBER OF DESIGN VARIABLE
C      NRESP : NUMBER OF RESPONSE
C
C*****
C      ***
C
      SUBROUTINE DAVESTA(DVARS,NDV,DAVE,DSDEV)
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      DOUBLE PRECISION DVARS(NDV)
      DOUBLE PRECISION DAVE(NDV)
      DOUBLE PRECISION DSDEV(NDV)

C
C      ! COMPUTE AVERAGE
C
      AVE = 0.0D0

```

Special Design Features

```
C
      DO I = 1, NDV
        AVE = AVE + DVAR(I)
      END DO

C
      AVE = AVE/REAL(NDV)
      DAVG = 1.0D0/REAL(NDV)

C
      DO K=1,NDV
        DO J=1,NDV
          DAVE(J) = 0.0D0
        END DO
        DAVE(K) = 1.0D0

C
C      ! COMPUTE VARIATION
C
      VAR = 0.0D0
      DVAR = 0.0D0

C
      DO I = 1, NDV
        VAR = VAR + (AVE - DVAR(I))**2/REAL(NDV)
        DVAR = DVAR +
&          2.0D0*(AVE - DVAR(I))*(DAVG - DAVE(I))/REAL(NDV)
      END DO

C
C      ! COMPUTE STANDARD DEVIATION
C
      SDEV = DSQRT(VAR)
      IF (SDEV.NE.0.0D0) THEN
        DSDEV(K) = 0.5D0/SDEV*DVAR
      ELSE
        DSDEV(K) = 0.0D0
      ENDIF

C
      END DO

C
      DO K=1,NDV
        DAVE(K) = DAVG
      END DO
      RETURN
      END
```

4.6.4 Use of the DRESP3 / TRESP3 capability

The **DRESP3** (**TRESP3**) input data has the same form as the **DRESP2** (**TRESP2**) input data except that the equation ID (EQID) is replaced with the interface library ID or a built-in equation NAME. There are interface subroutines available or 26 built-in equations are available: AVG1, AVG2, AVG3, AVG4, MULT1, MULT2, MULT3, MULT4, NORM1, NORM2, NORM3, NORM4, SUM1, SUM2, SUM3, SUM4, SAVG1, SAVG2, SAVG3, SAVG4, ASAVG1, ASAVG2, ASAVG3, ASAVG4, PNORM2 and SDEV.

The format of the DRESP3 data is:

1	2	3	4	5	6	7	8	9	10
DRESP3	ID	LABEL	LIBID or NAME						
+	"DVAR"	NDV1	NDV2	NDV3	NDV4	NDV5	NDV6	NDV7	NDV8
+		NDV9	NDV10	...					
+	"DTABLE"	NC1	NC2	...					
+	"DGRID"	NG1	NGC1	NG2	NGC2	...			
+	"DRESP1"	NR1	NR2	...					

The alternate formats are

1	2	3	4	5	6	7	8	9	10
DRESP3	ID	LABEL	LIBID or NAME						
+	"DVAR"	NDV1	NDV2	NDV3	NDV4	NDV5	NDV6	NDV7	NDV8
+		NDV9	NDV10	...					
+	"DTABLE"	NC1	NC2	...					
+	"DGRID"	NG1	NGC1	NG2	NGC2	...			
+	"DRESP1L"	NR1	LIDR1	NR2	LIDR2	

or:

1	2	3	4	5	6	7	8	9	10
DRESP3	ID	LABEL	LIBID or NAME						
+	"DVAR"	NDV1	NDV2	NDV3	NDV4	NDV5	NDV6	NDV7	NDV8
+		NDV9	NDV10	...					
+	"DTABLE"	NC1	NC2	...					
+	"DGRID"	NG1	NGC1	NG2	NGC2	...			
+	"DRESP2"	NS1	NS2	...					

The DRESP3 ID must be unique with respect to other DRESP3, DRESP1, DRESP2 and DRESPU ID's. Note that DRESP1L data can be used in the DRESP3 data.

Special Design Features

The DRESP3 responses are constrained with the DCONS data in the same manner as the DRESP2 responses. A DRESP3 response is specified as the objective function with the DOBJ / DMATCH / DINDEX data in the same manner as a DRESP2 response.

The format of the TRESP3 data is:

1	2	3	4	5	6	7	8	9	10
TRESP3	ID	LABEL	LIBID or NAME						
+	"TVAR"	NDV1	NDV2	NDV3	NDV4	NDV5	NDV6	NDV7	NDV8
+		NDV9	NDV10	...					
+	"DTABLE"	NC1	NC2	...					
+	"TRESP1"	NR1	NR2	...					

The alternate formats are

1	2	3	4	5	6	7	8	9	10
TRESP3	ID	LABEL	LIBID or NAME						
+	"TVAR"	NDV1	NDV2	NDV3	NDV4	NDV5	NDV6	NDV7	NDV8
+		NDV9	NDV10	...					
+	"DTABLE"	NC1	NC2	...					
+	"TRESP1L"	NR1	LIDR1	NR2	LIDR2	

or:

1	2	3	4	5	6	7	8	9	10
TRESP3	ID	LABEL	LIBID or NAME						
+	"TVAR"	NDV1	NDV2	NDV3	NDV4	NDV5	NDV6	NDV7	NDV8
+		NDV9	NDV10	...					
+	"DTABLE"	NC1	NC2	...					
+	"TRESP2"	NS1	NS2	...					

The TRESP3 ID must be unique with respect to other TRESP3, TRESP1 and TRESP2 ID's. Note that TRESP1L data can be used in the TRESP3 data.

The TRESP3 responses are constrained with the TCONS data in the same manner as the TRESP2 responses. A TRESP3 response is specified as the objective function with the TOBJ / TINIDEX data in the same manner as a TRESP2 response.

4.6.5 DRESP3 Interface Routine for DRESP3 / TRESP3 entries

The user must write a subroutine that, given the arguments, calculates the response value. This function must be included in a shared object (DLL) that is specified by the **DRESP3** executive control command. The Fortran name of the subroutine is DRESP3, and is declared as follows:

```
SUBROUTINE DRESP3(IWHICH, VAR, N, VAL, IERR)
  INTEGER IWHICH, N, IERR
  DOUBLE PRECISION VAR(N)
  DOUBLE PRECISION VAL
```

IWHICH is input and is the library ID specified on the **DRESP3** or **TRESP3** input data. This can be used to make a single shared object able to calculate multiple different user responses. VAR is input and contains the design variable values, table constants, grid point locations, and structural responses in the same order that they are specified in the DRESP3 or TRESP3 bulk data entry. N is input and contains the number of arguments in VAR. VAL is output and should contain the response value. IERR is output and should contain 0 if no errors were detected. If an error is encountered in this routine the error switch IERR should be set to 1, causing the program to terminate. The DRESP3 routine should set IERR = 2 if IWHICH corresponds to any value for which no user response has been defined.

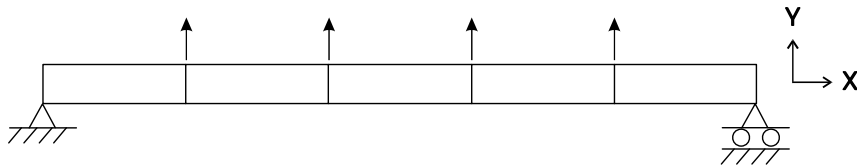
Note that the subroutine must always be named **DRESP3**, even if it is referenced by TRESP3 bulk data entries.

If a language other than Fortran is used to create the shared object, care must be taken to ensure that the interface function has the correct name and arguments. The actual required function names are system dependent. For example, using the C language, the DRESP3 interface function should have the following prototype:

Microsoft Windows	<code>__declspec(dllexport) void DRESP3(int *iwhich, double var[], int *n, double *val, int *ierr);</code>
Solaris, Linux, HP-UX	<code>void dresp3_(int *iwhich, double var[], int *n, double *val, int *ierr);</code>
AIX	<code>void dresp3(int *iwhich, double var[], int *n, double *val, int *ierr);</code>

4.6.6 Example with DRESP3 User Program

Consider the shape and sizing optimization of the simply supported beam shown below. The design variables control beam cross sectional dimensions and grid point locations. The objective function to be minimized is the mass of the beam. The constraint on the design is that the deformed shape of the beam be within 5% of a least squares fit of the first term of a Sine series.



A portion of the input data is shown below. The DRESP3 data consists of the X and Y locations and the U and V displacements of the six grid points. A total of 24 values are sent to the user program. The fourth field of the DRESP3 data specifies that the interface subroutine ID 1 is requested. The DCONS data specifies that the calculated value will be within 5% of the least squares fit.

	1	2	3	4	5	6	7	8	9	10
\$ DISPLACEMENT RESPONSES										
DRESP1	1	U1	DISP				1	1		
DRESP1	2	V1	DISP				2	1		
DRESP1	3	U2	DISP				1	2		
DRESP1	4	V2	DISP				2	2		
DRESP1	5	U3	DISP				1	3		
DRESP1	6	V3	DISP				2	3		
DRESP1	7	U4	DISP				1	4		
DRESP1	8	V4	DISP				2	4		
DRESP1	9	U5	DISP				1	5		
DRESP1	10	V5	DISP				2	5		
DRESP1	11	U6	DISP				1	6		
DRESP1	12	V6	DISP				2	6		
\$										
DRESP3	100	TERM1	1							
+	DGRID	1	1	1	2	2	1	2	2	
+		3	1	3	2	4	1	4	2	
+		5	1	5	2	6	1	6	2	
+	DRESP1	1	2	3	4	5	6	7	8	
+		9	10	11	12					
DCONS	100		1.05	0.95						

Special Design Features

The interface subroutines are shown below. First the deformed shape of the beam is determined by adding the coordinates of the grid points to their displacements. Then a least squares fit on a 1 term Sine series is performed. Finally the least squares term is stored in the variable VAL and returned.

```

      SUBROUTINE DRESP3(IWHICH,VAR,N,VAL,IERR)
      IMPLICIT NONE
C
      INTEGER IWHICH, N, IERR
C
      DOUBLE PRECISION VAR(N)
C
      DOUBLE PRECISION VAL
C
      CALL MYSUB1(VAR,N,VAL,IERR)

      RETURN
      END

      SUBROUTINE MYSUB1(VAR,N,VAL,IERR)
C
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
C
      DOUBLE PRECISION VAR(N)
C
      DOUBLE PRECISION X(6), Y(6)
C
      DATA X/0.0D0,0.0D0,0.0D0,0.0D0,0.0D0,0.0D0/
      DATA Y/0.0D0,0.0D0,0.0D0,0.0D0,0.0D0,0.0D0/
      DATA PI/3.141590D0/
      DATA RL/100.0D0/
C
      IERR = 0
C
      J = 0
      K = 12
C
      1. CALCULATE DEFORMED SHAPE = COORDINATES + DISPLACEMENTS
C
      DO 10 I = 1,6
      J = J + 1
      K = K + 1
      X(I) = VAR(J) + VAR(K)
      J = J + 1
      K = K + 1
      Y(I) = VAR(J) + VAR(K)
10    CONTINUE
C
      2. LEAST SQUARES FIT OF 1 TERM SINE SERIES
C
      T1 = 0.0D0
```



```

T2 = 0.0D0
T3 = 0.0D0
T4 = 0.0D0
T5 = 0.0D0
T6 = 0.0D0
DO 20 I = 1,6
    T1 = T1 + Y(I)**2
    T2 = T2 + SIN(PI*X(I)/RL)**2
    T3 = T3 + SIN(2.0D0*PI*X(I)/RL)**2
    T4 = T4 + 2.0D0*Y(I)*SIN(PI*X(I)/RL)
    T5 = T5 + 2.0D0*Y(I)*SIN(2.0D0*PI*X(I)/RL)
    T6 = T6 + 2.0D0*SIN(PI*X(I)/RL)*SIN(2.0D0*PI*X(I)/RL)
20 CONTINUE
C
    B1 = T5 - (2.0D0*T4*T3)/T6
    B2 = T6 - (4.0D0*T2*T3)/T6
    A1 = B1/B2
C
C    3. STORE IN VAL
C
    VAL = A1
C
    RETURN
END

```

4.6.7 DRESP3 / TRESP3 Built-in Functions

There are 26 built-in functions available. They are:

AVG1, AVG2, AVG3, AVG4, MULT1, MULT2, MULT3, MULT4, NORM1, NORM2, NORM3, NORM4, SUM1, SUM2, SUM3, SUM4, SAVG1, SAVG2, SAVG3, SAVG4, ASAVG1, ASAVG2, ASAVG3, ASAVG4, PNORM2 and SDEV.

These are defined as follows:

$$AVG1 = (X1 + X2 + \dots + XN)/N$$

$$AVG2 = (X1^{**2} + X2^{**2} + \dots + XN^{**2})/N$$

$$AVG3 = (X1^{**3} + X2^{**3} + \dots + XN^{**3})/N$$

$$AVG4 = (X1^{**4} + X2^{**4} + \dots + XN^{**4})/N$$

$$MULT1 = X1 * X2 * \dots * XN$$

$$MULT2 = MULT1^{**2}$$

$$MULT3 = MULT1^{**3}$$

$$MULT4 = MULT1^{**4}$$

$$NORM1 = (ABS(X1) + ABS(X2) + \dots + ABS(XN))$$

$$NORM2 = (ABS(X1)^{**2} + ABS(X2)^{**2} + \dots + ABS(XN)^{**2})^{** (1/2)}$$

$$NORM3 = (ABS(X1)^{**3} + ABS(X2)^{**3} + \dots + ABS(XN)^{**3})^{** (1/3)}$$

$$NORM4 = (ABS(X1)^{**4} + ABS(X2)^{**4} + \dots + ABS(XN)^{**4})^{** (1/4)}$$

$$SUM1 = X1 + X2 + \dots + XN$$

$$SUM2 = X1^{**2} + X2^{**2} + \dots + XN^{**2}$$

$$SUM3 = X1^{**3} + X2^{**3} + \dots + XN^{**3}$$

$$SUM4 = X1^{**4} + X2^{**4} + \dots + XN^{**4}$$

$$SAVG1 = ((X1-LB1)/(UB1-LB1) + (X2-LB2)/(UB2-LB2) + \dots + (XN-LBN)/(UBN-LBN))/N$$

$$SAVG2 = (((X1-LB1)/(UB1-LB1))^{**2} + ((X2-LB2)/(UB2-LB2))^{**2} + \dots + (XN-LBN)/(UBN-LBN))^{**2}/N$$

$$SAVG3 = (((X1-LB1)/(UB1-LB1))^{**3} + ((X2-LB2)/(UB2-LB2))^{**3} + \dots + (XN-LBN)/(UBN-LBN))^{**3}/N$$

$$SAVG4 = (((X1-LB1)/(UB1-LB1))^{**4} + ((X2-LB2)/(UB2-LB2))^{**4} + \dots + (XN-LBN)/(UBN-LBN))^{**4}/N$$

$$ASAVG1 = (ABS(X1)/MAX(ABS(UB1),ABS(LB1)) + ABS(X2)/MAX(ABS(UB2),ABS(LB2)) + \dots + ABS(XN)/MAX(ABS(UBN),ABS(LBN)))/N$$

$$ASAVG2 = ((ABS(X1)/MAX(ABS(UB1),ABS(LB1)))^{**2} + (ABS(X2)/MAX(ABS(UB2),ABS(LB2)))^{**2} + \dots + (ABS(XN)/MAX(ABS(UBN),ABS(LBN)))^{**2})/N$$

$$\text{ASAVG3} = ((\text{ABS}(\text{X1})/\text{MAX}(\text{ABS}(\text{UB1}),\text{ABS}(\text{LB1})))^{**3} + (\text{ABS}(\text{X2})/\text{MAX}(\text{ABS}(\text{UB2}),\text{ABS}(\text{LB2})))^{**3} + \dots + (\text{ABS}(\text{XN})/\text{MAX}(\text{ABS}(\text{UBN}),\text{ABS}(\text{LBN})))^{**3})/\text{N}$$

$$\text{ASAVG4} = ((\text{ABS}(\text{X1})/\text{MAX}(\text{ABS}(\text{UB1}),\text{ABS}(\text{LB1})))^{**4} + (\text{ABS}(\text{X2})/\text{MAX}(\text{ABS}(\text{UB2}),\text{ABS}(\text{LB2})))^{**4} + \dots + (\text{ABS}(\text{XN})/\text{MAX}(\text{ABS}(\text{UBN}),\text{ABS}(\text{LBN})))^{**4})/\text{N}$$

$$\text{PNORM2} = \text{SQRT} (\text{MAX}(\text{X1},0.0)^{**2} + \text{MAX}(\text{X2},0.0)^{**2} + \dots + \text{MAX}(\text{XN},0.0)^{**2})$$

$$\text{SDEV} = \text{SQRT} (((\text{X1}-\text{AVG1})^{**2} + (\text{X2}-\text{AVG1})^{**2} + \dots + (\text{XN}-\text{AVG1})^{**2})/\text{N})$$

Where, Xi corresponds to the listed values in the DVAR, DTABLE, DGRID and DRESP1 / DRESP2 lists in the DRESP3 entry or the TVAR, DTABLE and TRESP1 / TRESP2 lists in the TRESP3 entry, and N is the total number of listed arguments. LBi and UBi are the corresponding lower bound and upper bound from the DVAR entry for DVAR arguments, the corresponding lower bound and upper bound from the TVAR entry for TVAR arguments or LBi = 0.0 and UBi = 1.0 for all other arguments types.

4.7 Shifted Responses

Shifted responses are customized responses that combine values calculated from frequency response loadcases with user functions of the loading frequency. Shifted responses are used to easily create constraints on frequency response results where the constraint bound is a function of the loading frequency. Shifted responses requires using **DSHIFT**, **DRESP1** and **TABLEDx** entries. Shifted responses can include the user loading frequency function by either subtracting or normalizing the standard response. Shifted responses can also be use to change scales such as decibels.

Currently, *GENESIS* supports 15 shifted responses, whic are listed in the following table.

Shifted Responses types

Shifted Response Name	Shifted Response RTYPE
Shifted displacement from direct loadcases	DDISPS
Shifted velocity from direct loadcases	DVELOS
Shifted acceleration from direct loadcases	DACCES
Shifted displacement from modal loadcases	MDISPS
Shifted velocity from modal loadcases	MVELOS
Shifted acceleration from modal loadcases	MACCES
Shifted Random PSD displacement	PSDDS
Shifted Random PSD velocity	PSDVS
Shifted Random PSD acceleration	PSDAS
Shifted user function of displacement	UFDISPS
Shifted user function of velocity	UFVELOS
Shifted user function of acceleration	UFACCES
Shifted equivalenet radiated power	ERPS
Shifted dynamic element stress	DSTRS
Shifted dynamic element strain	DSTNS
Shifted dynamic element force	DFORCES

For example:

$$ACCES(f) = 20\log_{10}\left(\frac{ACCE(f)}{0.02}\right) - T(f)$$

In the example above, the acceleration is transformed to a decibel scale and shifted by $T(f)$.

4.7.1 Overview

The **DSHIFT** entry provides ways to define attributes to a standard frequency response listed in DRESP1.

The information on an entry:

1. Provides an ID so that one or more **DRESP1** entries can reference it.
2. Defines a scalar factor to scale the response (COEFF)
3. Defines a function (IDENT, LOG10 or LOG) to transform the response before applying the shift.
4. Defines a real number to normalize the response (REFVAL)
5. References a Table ID (TID) corresponding to a TABLED1, TABLED2, TABLED3 or TABLED4 entry that defines the function of the loading frequency.
6. Defines the shift type (ETYPE) of subtraction or normalization.

4.7.2 Data Entry

The format of the **DSHIFT** entry is presented next.

Format:

1	2	3	4	5	6	7	8	9	10
DSHIFT	ID	COEFF	FTYPE	REFVAL	TID	ETYPE			

Example 1: Create a shifted response, where tabled values are used to shift the response.

$$ACCES(f) = 20 \log_{10} \left(\frac{ACCE(f)}{0.02} \right) - T(f) :$$

1	2	3	4	5	6	7	8	9	10
DSHIFT	2222	20.0	LOG10	0.02	20001	1			

In this case a DRESP1 such the following should exist:

1	2	3	4	5	6	7	8	9	10
DRESP1	702	Z_DISP	MACCES			3	2222	55	

4.7.3 Loading Frequency Dependent Constraint Bounds

The **DSHIFT** entry provides ways to define a constraint bound that is a function of the the loading frequency.

Suppose that you have to impose the following constraint:

$$20\log_{10}\left(\frac{ACCE(f)}{0.02}\right) \leq T(f)$$

Because DCONS/DCONS2 only allow constant bounds, the above equation can not be used directly. However, we can write an equivalent equation as follows:

$$20\log_{10}\left(\frac{ACCE(f)}{0.02}\right) - T(f) \leq 0.0$$

In this case the bound is 0.0 which is constant and a constraint using and associated entries can be used:

	1	2	3	4	5	6	7	8	9	10
DSHIFT	2220	20.0	LOG10	0.02	3000	1				
DRESP1	700	Z_DISP	DISP			3	2220	55		
DCONS	700			0.0						

Alternatively, the equation can be rewritten as follows:

$$\frac{20\log_{10}\left(\frac{ACCE(f)}{0.02}\right)}{T(f)} \leq 1.0$$

In this case the bound is 1.0 which is also a constant and a constraint using and associated entries can be used:

	1	2	3	4	5	6	7	8	9	10
DSHIFT	2221	20.0	LOG10	0.02	3000	2				
DRESP1	701	Z_DISP	DISP			3	2221	55		
DCONS	701			1.0						

In the two alternative cases above, the desired constraint bounds $T(f)$ are assumed to be listed in a TABLED1 entry with ID 3000. The TABLED1entry could be:

	1	2	3	4	5	6	7	8	9	10
TABLED1	3000									
+	0.0	10.0	100.0	20.0	300.0	10.0	ENDT			

4.8 Restart Capability

The restart capability allows the user to restart from any previous design cycle. The last analysis is repeated, but the user can change the input data. The only restriction on the input data for the restart is that the number and order of the design variables remains constant. The restart uses the *pname.HIS* file that is created after every run.

4.8.1 Restart from any Previous Design Cycle

The user may want to restart from a previous design cycle. One reason could be that it is noticed after a particular design cycle that a design constraint or load case is missing. The user may also want to restart from the last design cycle.

To restart from any previous design the command **RESTART**=n,m needs to be used. This causes the design process to restart at design cycle “n”. The value of “m” is the new maximum number of design cycles. Note that this is not the additional number of design cycles. If “m” is not present then the value of DESMAX on the DOPT data (or the default value) is used. If “m” is present it must be greater than the original maximum number of design cycles. To restart from the last design cycle, the **RESTART**=LAST,k format of the command can be used. In this case, “k” is the maximum number of additional design cycles allowed.

The restart gets the design variable values from the *pname*.HIS file. The **HIS** command can be used to copy a history file from another run to *pname*.HIS at the start of the run. Certain input data (DCONS2, DINDEX, TCONS2, TINDEX) requires auxiliary restart information to be read from the *pname*.RST file. The **RST** command can be used to copy an RST file from another run to *pname*.RST at the start of the run.

All of the input data can be changed but the number and order of the design variables must remain constant. For example, to restart from the fifth design cycle the executive section of the input data would be:

```
ID CRANK
RESTART=5
CEND
```

4.9 Mode Tracking

Natural Frequency Loadcases

As the design changes throughout the design process, the order of the mode shapes can change. For example, initially the first bending mode of a structure could be at 7 Hz and the first torsional mode could be at 10 Hz. After the design is modified the first bending mode could be 11 Hz and the first torsional mode could be at 9 Hz. If a constraint was originally placed on the first bending mode, it would now be on the first torsional mode. Using mode tracking *GENESIS* can reorder the modes so that the constraint will remain on the first bending mode.

Mode tracking is enabled using the solution control command MODTRK. MODTRK can have the values of NO, YES, and ALL. NO is the default and does not enable mode tracking. YES will cause the frequencies or modes that are constrained to be tracked. ALL will cause all frequencies to be tracked. Another option to enable mode tracking is to use the parameter MODTRK. This parameter serves as a default for all eigenvalue loadcases that do not use the solution control command MODTRK.

If a constrained mode that is being tracked is not found during a design cycle analysis, then a fatal error will be issued and *GENESIS* will stop.

If a constrained mode that is being tracked may have switched with another mode, then a warning message will be issued after design convergence. In this case the user should check the results of the last analysis to make sure that the correct mode shape was constrained.

If modes switch position during the design process the frequencies and mode shapes in the output and postprocessing files will not be in order of increasing frequency.

Mode tracking information will be printed in the analysis results section of the output file. It is important to examine this data if restarts from the “.HIS” file are going to be used. This is because the mode numbers in the DRESP1 data will have to be changed if the mode switched order. The mode number in the design cycle to be restarted from will be the position of the mode when all of the modes are ordered from lowest to highest frequency.

It is typically needed to use mode tracking, whenever frequencies or eigenvector components are referenced by DRESP1.

Buckling Loadcases

Mode tracking can also be used to track modes associated to buckling analysis. The use of mode tracking in this case is very similar that with a natural frequency analysis and do not need further explanation as the same solution control commands: MODTRK and the same parameter, MODTRK, can be used.

Special Design Features

In buckling optimization, however, it is not typically needed to use mode tracking as in most cases the key is to either raise the lowest buckling load factor or constraint it, and individual modes are not as important.

4.10 Matching Measured Data

Sometimes, it is desirable to “tune” the analysis model so the results agree with those measured in an experiment or operating structure. For example, you may wish to match the calculated eigenvalues, some displacements and perhaps some stresses with those measured in an experiment. Sometimes, this is done by using a least squares fit to the responses in terms of member section properties. However, this can lead to unrealistic results, such as negative properties.

With *GENESIS*, you can still match calculated responses with desired or measured responses in a least squares sense. You can change any dimension, property or shape which is normally available in *GENESIS* for design optimization. Also, you can provide bounds on the parameters you wish to change and can even impose constraints on any responses, just as if you were designing the structure.

Analysis matching is performed by using the DMATCH data statement to define the design objective, and providing the data which you wish to have *GENESIS* match.

There are two methods available for analysis matching. The first is the Least Squares method (DOPT parameter IMATCH = 0, which is the default) and the second is the Beta method (DOPT parameter IMATCH = 1).

In the least squares method, the objective function is the sum of the squared, normalized, differences between the actual responses and those calculated by the finite element analysis. This can be stated as;

$$F = \sum_{i=1}^{NR} [M_i(R_i - T_i)]^2 \quad (\text{Eq. 4-2})$$

where T_i is the target value of the response, R_i is the calculated value, NR is the number of responses to be matched and M_i is a multiplier calculated using the following expression:

$$\begin{aligned} M_i &= \frac{W_i}{\text{Max}(|T_i|, \text{RMATCH})} && \text{if}(W_i > 0.0) \\ M_i &= |W_i| && \text{if}(W_i < 0.0) \\ M_i &= \frac{1.0}{\text{Max}(|T_i|, \text{RMATCH})} && \text{if}(W_i = \text{blank}) \end{aligned} \quad (\text{Eq. 4-3})$$

in the above expressions, W_i is a weighting factor specified in **DMATCH2** (**DMATCH** sets all W_i to 1.0) and the **DOPT** parameter **RMATCH** is 0.01 by default.

Constraints on the other responses can also be included in the optimization problem formulation

Special Design Features

In the Beta method, the objective function is the internal variable called BETA. An internal set of constraints is added to the original constraints to force the matching. The internal problem is;

$$\text{Minimize } \beta \quad (\text{Eq. 4-4})$$

$$\text{S.T. } -\beta \leq M_i(R_i - T_i) \leq \beta \quad i = 1, NR \quad (\text{Eq. 4-5})$$

In addition, all of the originally defined constraints are included, though they are not listed here.

4.11 Matching Eigenvector Components

To match eigenvector components, it is recommended to avoid the MAX norm (in the EIGR statement). The reason for this is that this norm is, in general, not continuous due to changes of the location of the maximum component of the eigenvector. Instead it is recommended to use any other norm (MAX0, MASS or POINT) because these are continuous norms.

It is strongly recommended to use mode tracking when trying to match eigenvector components. In other words use: MODTRK=YES or MODTRK=ALL in the natural frequency loadcase.

The method used to match eigenvector components uses the sensitivities (derivatives) of the eigenvectors. In *GENESIS*, Nelson's method is used to calculate eigenvector sensitivities. This method is only valid for non-repeated eigenvalues. If the problem does have repeated eigenvalues, then *GENESIS* will give a warning message, but will not terminate. If a repetition occurs only in a few design cycles, and not in the last one, the solution will probably be acceptable, but the user is warned to check the results with special care.

4.12 Relative Constraint Bounds

The **DCONS2** / **TCONS2** entries allow definition of constraints with bounds that are defined relative to the response's initial value. For constraints defined for all loadcases (**LID1** = 'ALL'), different bound values are calculated for each loadcase. For multivalued responses (e.g., stress for every element in a property), a single upper bound (per loadcase) is calculated based on the maximum (closest to $+\infty$) response value and a single lower bound (per loadcase) is calculated based on the minimum (closest to $-\infty$) response value.

The use of relative constraint bounds with responses that may have negative initial values (e.g., displacement components) can lead to non-intuitive infeasible constraints. For example, using **LBF** = 0.8 and **UBF** = 1.2 with a response that has an initial value of -1.0 will lead to a lower bound that is greater than the upper bound. For this reason, care must be used when choosing scale factors for such responses. To set relative constraint bounds on the magnitude of an indefinite response (i.e., regardless of whether it is positive or negative), it is recommended to pass the response into a **DRESP3** / **TRESP3** with the **NORM1** function to calculate the absolute value.

4.13 Allowable Probability of Failure on Constraint Entries

The **DCONS** / **DCONS2** entries allow the specification of allowable probability of failure (PFI) values.

In a typical reliability optimization problem, the program will first perform deterministic optimization and after converging to a deterministic answer, it will automatically switch to probabilistic optimization. In the deterministic mode, the probability of failure in the constraint bounds are not used. The DOPT parameter **RBOOST** can be used to set a limit on the number of deterministic design cycles performed before switching to probabilistic optimization.

4.14 Mesh Smoothing

Automatic mesh smoothing can be performed on models of 2D elements (QUAD4, TRIA3, QUAD8 and TRIA6) that are approximately planar and on models of solid elements (HEXA, PENTA, TETRA, PYRA and HEX20). This feature is useful when there are shape design variables as the mesh may become distorted. Automatic mesh smoothing is controlled by the analysis PARAMeter MSMOOTH which can have the values of ON, OFF (the default).

The mesh is smoothed at the end of each design cycle if mesh smoothing is turned on . The elements are smoothed property by property. The location of the grid points at the edges and boundary of the structure are not affected. The locations of the internal grid points are adjusted so as to minimize the element distortion of all the elements.

The element connectivity for 2D planar models should be consistent, i.e. clockwise or counter clockwise throughout the model. This will be automatically checked and corrected if necessary.

Statistics about the element distortion levels during the smoothing process can be obtained using DIAGnostics 961.

To output the changes due to mesh smoothing, use the Solution Control Command SHAPE = POST and the analysis PARAMeter PSMOOTH = ON.

4.15 Stress Ratio

The stress ratio method is an optimization method applicable to a special subset of structural optimization problems. It is used to resize members of a structure such that certain static stress constraints are exactly satisfied. In *GENESIS*, this method is typically used in conjunction with the standard method. When selected, the stress ratio method is used for the first ISRMAX design cycles and after that *GENESIS* will use the standard approximation method. ISRMAX is a DOPT parameter with a default value of 3. With the stress ratio method, *GENESIS* can be used to design areas of rods and/or thicknesses of shells or composites subject to stress and/or failure index constraints. In problems that contain other constraints, *GENESIS* will temporarily ignore them while using the stress ratio method. They will be considered when the approximation method gets activated.

Due to its limited scope, this method is not recommended for most problems. This method should only be used on the rare occasion when the standard method has problems resizing elements.

The advantage of the stress ratio method over the standard method is that it does not require the calculation of sensitivities or the solution of the approximate problem. This means that each design cycle can be faster.

It is interesting to note that the stress ratio method, when used to optimize a statically determinate structure that only has stress constraints, can converge in one design cycle. In non-statically determinate structures this is usually not true.

If this method is used alone, it may not converge to an optimal solution. The reason for this is that an optimal solution usually does not require that all members be fully stressed in at least one load case.

To update areas of rods, the stress ratio method uses the following equation:

$$\text{Area new} = \text{MAX}(\text{Area old} * \text{Stress} / \text{Stress limit}, \text{PMIN})$$

To update thickness of plates the stress ratio method uses the following equation:

$$\text{thickness new} = \text{MAX}(\text{thickness old} * \text{Stress} / \text{Stress limit}, \text{PMIN})$$

Where,

PMIN is the minimum property value as defined in DVPROP1.

The DOPT parameter, ISRMET, is used to control the stress ratio resizing method. If ISRMET=0, *GENESIS* will not use stress ratio (this is the default). If ISRMET = 1 or 3, *GENESIS* will not change the design variables that reference elements that do not have stress ratio constraints. The difference between 1 and 3 is that the later case will not stop due to hard convergence until at least one cycle using the standard approximation method has been completed. Between 1 and 3, 3 is recommended for most cases. Option 1 should be picked over 3 only when analysis is very time consuming. If ISRMET=2 or 4, *GENESIS* will attempt change the design variables associated to all stress ratio designable elements (e.g., shell elements without stress

Special Design Features

constraints will go to their minimum gage). The difference between 2 and 4 is that the later case will not stop due to hard convergence until at least one cycle using the standard approximation method has been completed. Between 2 and 4, 4 is recommended for most cases. Option 2 should be pick over 4 only when analysis is very time consuming.

4.16 Customization Through Scripts

GENESIS has the ability to load and run custom program instructions called scripts. *GENESIS* contains a built-in Lua scripting engine to interpret and execute the script lines. Scripts are loaded using the **SCRIPT** executive control command.

The format of the SCRIPT entry is as follows:

```
SCRIPT = engine, timeout
.
.
ENDSCRIPT
```

engine - The scripting engine. Currently, only the 'LUA' engine is available.

timeout - The script timeout (in secs). To prevent a script from falling into an infinite loop, all script function executions are timed. If any script function takes longer than timeout, the script engine will abort and no further script functions will run. (Integer > 0. Default = 900)

All lines between SCRIPT and ENDSCRIPT are interpreted by the scripting engine. If SCRIPT occurs inside an included file, and the end of that file is reached while reading script lines, an implicit ENDSCRIPT will be automatically inserted before continuing to read lines from the main file. Note that unlike normal input data, the scripting engine will interpret lines in a case-sensitive manner.

4

4.16.1 Lua scripting engine

GENESIS includes a Lua 5.3 interpreter. See <http://www.lua.org/> for documentation of lua syntax. Most of the standard lua libraries are pre-loaded. These include: `base`, `io`, `math`, `os`, `string` and `table`. Note: `package` is not available, and native code libraries cannot be loaded into *GENESIS* by a script.

`print()` and `io.write()` will print to the *GENESIS* output file.

`os.exit()` will raise a lua error, which will abort the *GENESIS* run.

`os.execute()` can be used to run external programs, and those programs will not be limited by *timeout*. Script processing will wait for the external program to finish, and may wait longer than *timeout*.

genesis table

GENESIS provides a `genesis` table in the global environment. This table has the following fields:

```
genesis.common
genesis.db
genesis.dcons
genesis.diag
genesis.dopt
genesis.dresp
genesis.dselect
genesis.dtable
genesis.hconv
genesis.hook
genesis.pname
genesis.rcons
genesis.sconv
genesis.tselect
genesis.version
```

Each of these is described in detail below:

genesis.common

Function taking one string argument. The string is the name of a *GENESIS* common block array. The function will return an object that can take an integer index to get/set the corresponding values in the *GENESIS* array.

The *GENESIS* common block arrays are:

```
"NC11A"
"NC12A" "NC12D" "NC12T" "RNC12R"
"NC14A" "NC14B" "NC17A" "NC17B"
"NC16D" "ITC16T"
"NC30A" "NC30B" "RNC30R" "NC37A" "NC37B" "RNC37R"
"NC41S"
```

Changing values in common block arrays will very likely cause unexpected results, and may even cause *GENESIS* to crash. Only alter common block arrays as advised by *GENESIS* support.

Example:

```
nc12a = genesis.common("NC12A")
print("ngrid = "..nc12a[2])
```

genesis.db

Function taking one string argument. The string is the name of a *GENESIS* database table. The function will return an object that can take an integer index to get/set the corresponding values in the *GENESIS* table. The object can also be indexed to call the following functions:

```
db.print([nrow])
db.insert(size)
db.delete()
y_db.fill(val [, n [, iy [, incy]]])
    y = val
y_db.fill(x_db [, n [, iy [, ix [, incy[, incx]]]]])
    y = x
y_db.scale(val [, n [, iy [, incy]]])
    y = val y
y_db.axpy(a, x_db [, n [, iy [, ix [, incy[, incx]]]]])
    y = a x + y
y_db.dotprod(x_db [, n [, iy [, ix [, incy[, incx]]]]])
    returns <x,y>
y_db.rand([n [, iy [, incy]]])
    y = (each component randomly scaled)
```

`insert()` and `delete()` can only be applied to tables reserved for lua scripts. These tables are named '#LUA*i*' *i*=0...9

The lua length operator (#) can be applied to *db* objects.

Note: There are potentially thousands of tables in the *GENESIS* database. Changing values in arbitrary tables will very likely cause unexpected results, and may even cause *GENESIS* to crash. Only manipulate tables as advised by *GENESIS* support.

Example:

```
genesis.db("IDGRID").print(2)
```

genesis.dcons

Function taking one or two integer arguments. The first is the ID of a **DRESP1**, **DRESP2**, **DRESP3** or **DRESPG** entry. The second is an optional loadcase ID. If the loadcase ID is omitted or 0, then all applicable loadcases are affected. The function will return an object that can take a string index of “lb” or “ub” to get/set the lower or upper bound, respectively, of the **DCONS/DCONS2** constraints. Note that this can only reset constraint bounds, and cannot create previously non-existing constraints. If the input data does not constrain the response for the given loadcase(s), attempts to reset the bounds will be ignored. Note that constraint bounds should not be changed between the DRESP hook and the SCONVG hook. In particular, constraint bounds should not be changed in SCREEN, PRINTD, PRE_HCONVG, HCONVG, SENS, SENSIT, SENPRT or APPROX.

See **genesis.hook**.

Example:

```
dc101 = genesis.dcons(101,1)
print("DCONS 101 lower bound = ",dc101.lb)
```

genesis.diag

Object that can be indexed with an integer (1-99) to get/set diagnostic switches. Note that in *GENESIS* input data DIAG, the most significant two digits sets the switch index, and the rest of the digits set the value: E.g., DIAG=324 is equivalent to `genesis.diag[32] = 4`. If the DIAG is only two digits, then the most significant digit sets the index and the other digit sets the value.

Example:

```
genesis.diag[1] = 2 -- same as DIAG=12
```

genesis.dopt

Object that can be indexed with a string to get/set DOPT parameters. The name DESMAX can be used to get/set the maximum allowable design cycles.

Example:

```
print("desmax = ",genesis.dopt.DESMAX)
```


genesis.dresp

Function that can be used to query DRESP1/2 values (*only* in the DRESP1 or DRESP hook functions. See [genesis.hook](#)).

This function can be called in one of four different styles:

```
genesis.dresp( resp [, lc] )
    returns two values, the maximum and minimum responses
genesis.dresp( resp, lc, "average", [amode [, anum]] )
    returns the average response value
genesis.dresp( resp, lc, "count", cside, ccut )
    returns the number of responses passing the cutoff
genesis.dresp( resp, lc, "fill", db, [fmode [, fnum]] )
    copies response values into db
```

Parameters:

resp - user DRESPi ID (integer)
lc - user loadcase ID (integer, default = 0 meaning all)
amode - one of "all", "top", "bottom" (string, default = "all")
anum - for *amode*="top" or "bottom", count of responses to include in calculation (integer <= 100)
cside - one of "<", ">" (string)
ccut - cutoff value (number)
db - A database table returned by `genesis.db()`. This table must be one reserved for lua scripts (i.e., named '#LUAi' *i*=0...9).
fmode - one of "top" or "bottom" (string, default = "top")
fnum - for *fmode*="top" or "bottom", count of responses to include (integer <= 100)

Examples:

```
rmax,rmin = genesis.dresp(1001)
nvio = genesis.dresp(2001,10,"count",>,100.0)
rval = genesis.dresp(2040,21,"average","top",10)
```

genesis.dselect

Function taking one integer argument, which is the ID of a **DSELECT** entry. The function will return an object that can take a string index of “lb” or “ub” to get/set the lower or upper bound, respectively, of the constraints internally created by the corresponding DSELECT. Note that if the DSELECT entry specifies a BTYPE of “UPPER”, then the lower bound does not exist and attempts to set it will be ignored.

Example:

```
ds101 = genesis.dselect(101)
print("DSELECT 101 lower bound = ",ds101.lb)
```

genesis.dtable

Object that can be indexed with a string to get/set DTABLE constant values. Note that DTABLE values should not be changed between the DRESP1 hook and the SCONVG hook. In particular, DTABLE values should not be changed in DRESP, SCREEN, PRINTD, PRE_HCONVG, HCONVG, SENS, SENSIT, SENPRT or APPROX.

See **genesis.hook**.

genesis.hconv

Integer flag that can be used to get/set hard convergence (*only* in the HCONVG hook function. See **genesis.hook**).

0 = not converged
1 = hard convergence
2 = max design cycles reached

Note that `genesis.hconv == 2` is a notification only. This value cannot be reset to avoid stopping *GENESIS*. Change `genesis.dopt.DESMAX` in the PRE_HCONVG hook to avoid stopping for max cycles.

genesis.hook

Table of hook functions. The script, as input, is run early on, before the bulk data is even read in. Therefore, anything "interesting" the script is going to do must be done via hook functions (callbacks). The script sets hook functions by assigning function values to the `genesis.hook` table. As *GENESIS* runs, at defined moments throughout each design cycle it will check the hook table for specific hook names. If a hook function exists, it will call that function, passing it three arguments: the current design cycle number, the current objective function value and the current maximum constraint violation. Note that the cycle number changes in the UPDATE module, while the objective and constraint violation values change in the SCREEN module.

GENESIS will call a hook at the end of each main module (with the module names as reported by the TIMES command). Additional hook moments are also defined.

List of hooks called once during data preprocessing:

```
'READ'      'PREPRO'  'HEATPR'  'AUTOGN'  'TOPRE1'  'TOPRE2'  'TOPRE3'
'DESIGN'    'SMOOTH'  'CHECK'   'PRINTI'  'PRINTU'  'ESL'
```

List of hooks called every design cycle:

```
'FEH'      'SOLV_H'  'FE'      'URMASS'  'SOLVER'  'STRESS'  'CSTRES'
'RANDOM'   'PRINTA'  'DRESP1'  'DRESP'   'SCREEN'  'PRINTD'  'PRE_HCONVG'
'HCONVG'   'SENSH'   'SENSIT'  'SENPRT'  'APPROX'  'SCONVG'  'UPDATE'
'SMOOTH'   'OPT'     'CHECK'   'PRINTU'
```

List of hooks called once during program finalization:

```
'TSURF'    'SSOL'    'AUTORIB' 'FINISH'
```

Note that some hooks may be skipped depending on input data. For example, the SENPRT hook is only called if the **SENSITIVITY** solution control command is used to request sensitivity printing.

Certain hooks are provided special abilities to access and/or change certain *GENESIS* data. These include:

`genesis.hook.DRESP1`

This hook can use the `genesis.dresp` function to query DRESP1 values and can use the `genesis.dtable` object to change DTABLE values before DRESP2 responses are evaluated. Note: querying DRESP2 responses from this hook will return garbage.

`genesis.hook.DRESP`

This hook can use the `genesis.dresp` function to query DRESP1/DRESP2 values. This could be used to create custom response histories, for example in a spreadsheet file.

`genesis.hook.PRE_HCONVG`

This hook provides an opportunity for the script to change `genesis.dopt.DESMAX` before testing for a max cycle stop. Note: the external *GENESIS* control program may change DESMAX after this hook runs to force a max cycle stop anyway.

`genesis.hook.HCONVG`

This hook can query the `genesis.hconv` field to determine whether *GENESIS* has met the hard convergence criteria and/or reset `genesis.hconv` to instruct *GENESIS* to stop or not stop. Note: max cycle stop cannot be overridden by this hook.

`genesis.hook.SCONVG`

This hook can query the `genesis.sconv` field to determine whether *GENESIS* has met the soft convergence criteria and/or reset `genesis.sconv` to instruct *GENESIS* to stop or not stop.

Example:

```
genesis.hook.SOLVER = function(cycle)
  print("In lua solver hook.")
  print("Solver finished for cycle ", cycle)
```

end

genesis.pname

String containing project name. This is normally the input filename (minus any extension).

genesis.rcons

Function that can be used to get information about retained constraints. Takes one integer parameter, which is the index of the retained constraint to be inquired.

`genesis.common("NC41S")[1]` gives the number of retained constraints.

The function returns 10 values:

g (number) -- the normalized constraint value
 r (number) -- the actual response value
 b (number) -- the bound value
 bt (int) -- the bound type (1 = upper bound, -1 = lower bound)
 rt (string) -- the response type
 lc (int) -- loadcase id or 0 if N/A
 fq (int) -- loading frequency number or mode number or 0 if N/A
 id (int) -- grid/elem/prop/mat/dresp id
 ic (int) -- item code
 j (int) -- column number in GRAD table

Example:

```
genesis.hook.SCREEN = function(cycle, obj, viomax)
  print("Retained constraints for cycle ", cycle)
  local nconr = genesis.common "NC41S" [1]
  for i=1,nconr do
    local g,r,b,bt,rt,lc,fq,id,ic,j = genesis.rcons(i)
    print(i,rt,lc,id,ic,r,b)
  end
end
```

genesis.sconv

Integer flag that can be used to get/set soft convergence (*only* in the SCONVG hook function. See [genesis.hook](#)).

0 = not converged
 1 = soft convergence

genesis.solc

Object that can be indexed with a string to get/set solution control switches. The following names can be used:

APRINT, DPRINT, UPRINT, OPRINT, MPRINT, MASS, VOLUME, SUMMARY, DRESP2, DESIGN, GRAPH, SENSITIVITY, SHAPE, SIZING, DENSITY, TVAR.

Example:

```
print("aprint = ",genesis.solc.APRINT)
```

genesis.tselect

Function taking one integer argument, which is the ID of a **TSELECT** entry. The function will return an object that can take a string index of “lb” or “ub” to get/set the lower or upper bound, respectively, of the constraints internally created by the corresponding TSELECT. Note that if the TSELECT entry specifies a BTYPE of “UPPER”, then the lower bound does not exist and attempts to set it will be ignored.

Example:

```
ts101 = genesis.tselect(101)
print("TSELECT 101 upper bound = ",ts101.ub)
```

genesis.version

String containing *GENESIS* version as "major.minor". Scripting features may change in future *GENESIS* versions. This can be used by a script to make sure that it is being run in a compatible *GENESIS* version.

xlsxwriter table

GENESIS may provide a `xlsxwriter` table in the global environment. This table provides access to methods that can write `xlsx`-formatted spreadsheet files.

`xlsxwriter.open(filename)`

Function taking one string argument. The string is the name of a spreadsheet file to create. The usual spreadsheet file extension is ".xlsx". The function will return a *workbook* object.

`workbook.add_worksheet(name)`

Function taking one string argument. The string is the unique name of a worksheet within the workbook. The function will return a *worksheet* object.

Set data within the worksheet with the following methods:

```
worksheet:write_number(row, col, number[, format])
worksheet:write_string(row, col, string[, format])
worksheet:write_formula(row, col, formula[, format])
worksheet:write_array_formula(row1, col1, row2, col2, formula[,
format])
worksheet:write_datetime(row, col, table[, format])
worksheet:write_url(row, col, url[, format])
worksheet:write_blank(row, col[, format])
worksheet:set_row(row[, height][, format])
worksheet:set_columns(col1, col2[, width][, format])
worksheet:merge_range(row1, col1, row2, col2, string[, format])
worksheet:insert_chart(chart, row, col)
worksheet:insert_image(filename, row, col)
```

`workbook.add_chart(type)`

Function taking one integer argument. The integer is the type of the chart. Constants below are built into the *workbook* object for the available types. The function will return a *chart* object that can be inserted into a *worksheet*.

The type argument should be one of the following:

```
workbook.CHART_AREA
workbook.CHART_AREA_STACKED
workbook.CHART_AREA_STACKED_PERCENT
workbook.CHART_BAR
workbook.CHART_BAR_STACKED
```

```

workbook.CHART_BAR_STACKED_PERCENT
workbook.CHART_COLUMN
workbook.CHART_COLUMN_STACKED
workbook.CHART_COLUMN_STACKED_PERCENT
workbook.CHART_DOUGHNUT
workbook.CHART_LINE
workbook.CHART_PIE
workbook.CHART_SCATTER
workbook.CHART_SCATTER_STRAIGHT
workbook.CHART_SCATTER_STRAIGHT_WITH_MARKERS
workbook.CHART_SCATTER_SMOOTH
workbook.CHART_SCATTER_SMOOTH_WITH_MARKERS
workbook.CHART_RADAR
workbook.CHART_RADAR_WITH_MARKERS
workbook.CHART_RADAR_FILLED

```

The *chart* object has the following methods:

```
chart:add_series(worksheet, row1, col1, row2, col2)
```

Returns a *series* object. The range [row1:row2,col1:col2] on *worksheet* sets the values for the series.

```
chart:set_title(string [,row ,col])
```

If row, col are given, then string must be the name of a worksheet. Otherwise string is the literal title.

```
chart:set_xaxis_label(string [,row ,col])
```

```
chart:set_yaxis_label(string [,row ,col])
```

```
chart:set_style(int)
```

```
chart:set_rotation(degrees)
```

```
chart:set_hole_size(int)
```

The *series* object has the following methods:

```
series:set_categories(worksheet, row1, col1, row2, col2)
```

```
series:set_name(string [,row ,col])
```

***workbook*:add_format()**

Function taking no argument. The function will return a *format* object that can be passed into *worksheet* methods to set the cell format.

Set the format characteristics by assigning one or more property keys on the *format* object.

Format object available property keys:

```
format.font = font_name -- string
```

```
format.size = size -- integer
```

```

format.color = format.COLOR_*
format.bold = true
format.italic = true
format.underline = format.UNDERLINE_*
format.strikeout = true
format.superscript = true
format.subscript = true
format.number_format = number_format -- string
format.number_format_index = index -- integer
format.align = format.ALIGN_*
format.wrap = true
format.rotation = angle -- integer [-90,90] or 270
format.indent = level -- integer
format.shrink = true
format.pattern = format.PATTERN_*
format.pattern_bg_color = format.COLOR_*
format.pattern_fg_color = format.COLOR_*
format.border = format.BORDER_*
format.border_bottom = format.BORDER_*
format.border_top = format.BORDER_*
format.border_left = format.BORDER_*
format.border_right = format.BORDER_*
format.border_color = format.COLOR_*
format.border_bottom_color = format.COLOR_*
format.border_top_color = format.COLOR_*
format.border_left_color = format.COLOR_*
format.border_right_color = format.COLOR_*

```

Format object available constants:

```

format.UNDERLINE_SINGLE
format.UNDERLINE_DOUBLE
format.UNDERLINE_SINGLE_ACCOUNTING
format.UNDERLINE_DOUBLE_ACCOUNTING
format.ALIGN_NONE
format.ALIGN_LEFT
format.ALIGN_CENTER
format.ALIGN_RIGHT
format.ALIGN_FILL
format.ALIGN_JUSTIFY
format.ALIGN_CENTER_ACROSS
format.ALIGN_DISTRIBUTED
format.ALIGN_VERTICAL_TOP

```


format.ALIGN_VERTICAL_BOTTOM
format.ALIGN_VERTICAL_CENTER
format.ALIGN_VERTICAL_JUSTIFY
format.ALIGN_VERTICAL_DISTRIBUTED
format.COLOR_BLACK
format.COLOR_BLUE
format.COLOR_BROWN
format.COLOR_CYAN
format.COLOR_GRAY
format.COLOR_GREEN
format.COLOR_LIME
format.COLOR_MAGENTA
format.COLOR_NAVY
format.COLOR_ORANGE
format.COLOR_PINK
format.COLOR_PURPLE
format.COLOR_RED
format.COLOR_SILVER
format.COLOR_WHITE
format.COLOR_YELLOW
format.PATTERN_NONE
format.PATTERN_SOLID
format.PATTERN_MEDIUM_GRAY
format.PATTERN_DARK_GRAY
format.PATTERN_LIGHT_GRAY
format.PATTERN_DARK_HORIZONTAL
format.PATTERN_DARK_VERTICAL
format.PATTERN_DARK_DOWN
format.PATTERN_DARK_UP
format.PATTERN_DARK_GRID
format.PATTERN_DARK_TRELLIS
format.PATTERN_LIGHT_HORIZONTAL
format.PATTERN_LIGHT_VERTICAL
format.PATTERN_LIGHT_DOWN
format.PATTERN_LIGHT_UP
format.PATTERN_LIGHT_GRID
format.PATTERN_LIGHT_TRELLIS
format.PATTERN_GRAY_125
format.PATTERN_GRAY_0625
format.BORDER_NONE
format.BORDER_THIN
format.BORDER_MEDIUM

```
format.BORDER_DASHED
format.BORDER_DOTTED
format.BORDER_THICK
format.BORDER_DOUBLE
format.BORDER_HAIR
format.BORDER_MEDIUM_DASHED
format.BORDER_DASH_DOT
format.BORDER_MEDIUM_DASH_DOT
format.BORDER_DASH_DOT_DOT
format.BORDER_MEDIUM_DASH_DOT_DOT
format.BORDER_SLANT_DASH_DOT
```

workbook:define_name(name, formula)

Function taking two string arguments. The first string is the name to define. The second string is the formula to assign to the name.

workbook:set_properties(props)

Function taking one table argument. The table defines the properties to set. This table can have keys with the following strings: "title", "subject", "author", "manager", "company", "category", "keywords", "comments" and/or "status". String values for these keys will be assigned to the corresponding property in the xlsx file.

workbook:close()

Function taking no argument. Will close spreadsheet file, writing out all data. Note that no data is written to the file until the workbook is closed. The workbook will not close automatically when *GENESIS* exits, so failing to explicitly close the workbook will result in no xlsx file being created.

CHAPTER 5

Topology Optimization

- Introduction
- Topology Design Bulk Data
- Topology Post Processing Result Files
- Using the AUTORIB capability with Topology Optimization
- Practical Recommendations
- Example
- Mixing Topology with Shape, Sizing, Topometry, Topography and/or Freeform Optimization

5.1 Introduction

Topology optimization is used to find the optimal distribution of material in a given package space. Unlike shape and sizing optimization, topology optimization does not require an initial design. Typically, the design starts with a block of material formed by a large number of finite elements and the topology optimization will “take out” from the block the unnecessary elements [9].

Topology optimization has a limited number of responses associated with it. These responses are primarily used to create a stiff and light structure.

Topology optimization can be performed simultaneously with shape/sizing optimization. By default, all topology and shape/sizing data in an input file will be used. If a problem has both a topology objective and a shape/sizing objective, they will be combined into an index objective function with equal weight factors. If a data file has both types of data the executive control command **TOPOLOGY** may be used to limit the optimization to using only the topology data or only the shape/sizing data.

Topology optimization is normally used by design engineers to perform conceptual designs. After the topology optimization is finished, shape and/or sizing optimization can be performed to refine the solution. To do the shape/sizing optimization the user has to re-build the analysis model by taking out the elements that topology has indicated to be unnecessary.

Topology optimization can also be performed simultaneously with shape, sizing and the other types of optimization. In this case, the data entries for all types of optimization can be used simultaneously.

In *GENESIS*, topology optimization works by creating design variables associated with the Young’s modulus and density of each element in the package space. The value of the design variable ranges between 0.0 and 1.0, where 1.0 indicates that the element has its normal stiffness and mass, and 0.0 indicates that the element has no stiffness or mass. Several different relationships between the stiffness and density of an element are available.

Topology optimization can be used with static, non-linear contact, eigenvalue, buckling, dynamic, random and heat transfer load cases. The most relevant results are the displacements, strain energy, grid contact clearance, contact pressure, natural frequency, buckling load factors, modal/direct/random displacement velocities, acceleration, temperature and heat transfer compliances (HTC) responses. The remaining analysis results (e.g., STRESS, STRAIN and FORCES) should only be used as reference solutions because they are theoretically valid only in the limits of the design variables (0.0 or 1.0). The reason for this is obvious: material properties are not really variable. This is just a method to identify which material to keep (design variable close to 1.0) and which material to discard (design variable close to 0.0).

Geometric responses such as moment of inertia, center of gravity and mass fraction can also be used in topology optimization.

Fabrication requirements such as minimum member size, maximum member size, castability, extrusion, stamping and symmetries can be imposed if desired.

5.2 Topology Design Bulk Data

The topology optimization data is used to:

1. Define design regions where the topology optimization can be applied to take out or keep material (TPROP).
2. Select the relevant responses (TRESP1, TRESP2, TRESP3,) to be used as constraints (TCONS, TCONS2) or the objective function (TOBJ, TINDEX).
3. Select optional symmetry planes and/or fabrication constraints(TSYM1, TSYM2 or TSYM3).
4. Select optional design regions that will be cloned from original design regions. (TPROPC).
5. Optionally define extra variables, constant values and equations to be used for synthetic responses (TVAR, DTABLE, DEQATN, TSELECT).
6. Optionally define a special shedule for the power rule method (TCYCLE).

The collection of the designable regions, planes of symmetry, constraints and objective function define the topology model. The topology design data is used to relate the analysis model with the topology design model.

Sizing, shape and other optimization types can be used simultaneously with topology optimization.

5.2.1 Topology Designable Elements

GENESIS can topologically design almost any element with property data that references an isotropic material (i.e., a MAT1 data statement). *GENESIS* can also topologically design solid elements that reference orthotropic or anisotropic material (MAT11 or MAT9), as well as elements that reference PCOMP/PCOMPG property data which can reference MAT2 and/or MAT8. In addition, heat transfer elements referencing MAT4 and MAT5 can also be designed. Following is a list of all elements designable with topology optimization:

TOPOLOGICALLY DESIGNABLE ELEMENTS		
Element	Property	Material
CROD	PROD	MAT1/ MAT4
CBAR	PBAR	MAT1/ MAT4
CBEAM	PBEAM	MAT1/ MAT4
CWELD	PWELD	MAT1
CTRIA3	PSHELL	MAT1/ MAT4/ MAT5
CQUAD4	PSHELL	MAT1/ MAT4/ MAT5
CTRIA6	PSHELL	MAT1/ MAT4/ MAT5
CQUAD8	PSHELL	MAT1/ MAT4/ MAT5
CTRIA3	PCOMP	MAT1 / MAT2/ MAT8 / MAT4/ MAT5
CQUAD4	PCOMP	MAT1 / MAT2/ MAT8 / MAT4/ MAT5
CTRIA6	PCOMP	MAT1 / MAT2/ MAT8 / MAT4/ MAT5
CQUAD8	PCOMP	MAT1 / MAT2/ MAT8 / MAT4/ MAT5
CSHEAR	PSHEAR	MAT1/ MAT4
CTRIAX6	PAXIS	MAT1/ MAT4/ MAT5
CTETRA	PSOLID	MAT1 / MAT9/ MAT11 / MAT4/MAT5
CPYRA	PSOLID	MAT1 / MAT9/ MAT11 / MAT4/MAT5
CPENTA	PSOLID	MAT1 / MAT9/ MAT11 / MAT4/MAT5
CHEXA	PSOLID	MAT1 / MAT9/ MAT11 / MAT4/MAT5
CHEX20	PSOLID	MAT1 / MAT9/ MAT11 / MAT4/MAT5

These element can be simultaneously designed. All other elements can be used in the model, but cannot be designed.

5.2.2 Topologically Designable Region Selection

To select a designable region the user needs to specify a group of elements. All elements referencing a given property ID are made designable with the **TPROP** data statement.

The basic format for TPROP is:

1	2	3	4	5	6	7	8	9	10
TPROP	ID	PTYPE	PID	INIT					

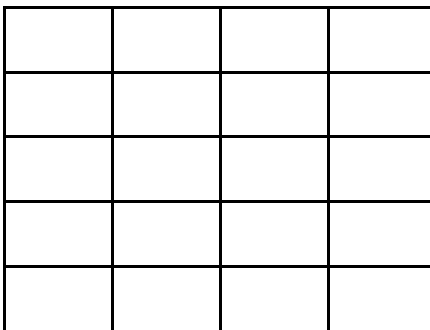
The ID is a unique identification number to identify the TPROP data. The ID is used by the program to report errors to the user. PTYPE is the property type. PID is the property identification number. All elements that reference PID will be designed. INIT is the initial value of the topology design variables.

Typically, INIT is defined to match the mass fraction constraint, so the initial design does not have violated constraints. For example, if the mass fraction is constrained to 40%, then it is suggested that $INIT = 0.4$

The topology design variables are created automatically by the program. Usually, there is one design variable per element. When symmetry planes are used normally there will be one design variable per pair of symmetric elements.

Example:

Consider the following mesh that contains 20 CQUAD4 element and each has the same property defined in PSHELL 100:



If the topology optimization task is to design each of the 20 elements, the following data could be created:

1	2	3	4	5	6	7	8	9	10
TPROP	1	PSHELL	100	0.4					

With this data *GENESIS* will internally create 20 design variables. Each of the design variables will have an initial value of 0.4.

Multiple TPROP entries are allowed in the input data. At most one TPROP entry may reference a given property ID.

5.2.3 Relationships Between Design Variables and Material Properties

GENESIS uses the density based method to solve the topology optimization problem. This method requires the creation of relationships between the design variables and the materials.

The typical relationship (POWER rule, which is the *GENESIS* default) is:

$$E(X) = E_0 RV2 + E_0(1 - RV2)X^{RV1} \quad (\text{Eq. 5-1})$$

$$\rho(X) = \rho_0 X \quad (\text{Eq. 5-2})$$

$$TMIN \leq X \leq 1.0 \quad (\text{Eq. 5-3})$$

where

$E(X)$ - Young's modulus

E_0 - Initial Young's modulus (this is the value in MAT1)

$\rho(X)$ - Density

ρ_0 - Initial density (this is the value in MAT1)

X - Topology design variables which represents the volume fraction (fraction of solid material)

$TMIN$ - Minimum value of the topology design variable.

$RV1$ - Real value supplied by user (Typically: $2.0 \leq RV1 \leq 3.0$)

$RV2$ - Real parameter representing $\frac{E_{MIN}}{E_0}$, where E_{MIN} is the minimum value

Young's modulus is allowed to take. ($0.0 < RV2 \leq 1.0$, Typically $RV2=10^{-6}$ which is the Default)

These equations create a heuristic relationship between the Young's modulus and the density. In theory the relationships are true only if the design variables are 0.0 or 1.0. If a design variables is 1.0 then what this means is that its corresponding element is needed. If the design variable is 0.0 then its corresponding element is not needed and therefore it can be taken out of the model. For numerical computation purposes, either $TMIN$ or $RV2$ must be provided (should be nonzero) to avoid singularity of the stiffness matrix.

The relationships above are not the only possible ones. The following relationships are also available in *GENESIS*:

$$E(X) = E_0 \left\{ RV1[X + RV2(1 - X)] + \frac{(1 - RV1)RV2}{(RV2 - 1)X + 1} \right\} \quad (\text{Eq. 5-4})$$

$$\rho(X) = \rho_0[X + RV3(1 - X)] \quad (\text{Eq. 5-5})$$

$$TMIN \leq X \leq 1.0 \quad (\text{Eq. 5-6})$$

where

$E(X)$ - Young's modulus

E_0 - Initial Young's modulus (this is the value in MAT1)

$\rho(X)$ - Density

ρ_0 - Initial density (this is the value in MAT1)

X - Topology design variable which represents the volume fraction.

$TMIN$ - Minimum value of the topology design variable.

$RV1$ - Hybridization parameter between Voigt ($RV1=1.0$) and Reuss ($RV1=0.0$) mixing. ($0.0 \leq RV1 \leq 1.0$) Default = 0.

$RV2$ - Real parameter representing $\frac{E_{MIN}}{E_0}$, where E_{MIN} is the minimum value Young's modulus is allowed to take. ($0.0 < RV2 \leq 1.0$, Typically $RV2=10^{-6}$ which is the Default)

$RV3$ - Real parameter representing $\frac{\rho_{MIN}}{\rho_0}$, ρ_{MIN} is the minimum value that the density is allowed to take. ($0.0 < RV3 \leq 1.0$, Typically $RV3=10^{-6}$ which is the Default)

These relationships are the "MIX" rule.

1. If $RV1 = 0.0$ (The Default for "Mix").

$$E(X) = E_0 \frac{RV2}{(RV2 - 1)X + 1} \quad (\text{Eq. 5-7})$$

$$\rho(X) = \rho_0[X + RV3(1 - X)] \quad (\text{Eq. 5-8})$$

This case corresponds to the Reuss mixing rule (isostress).

2. If $RV1 = 1.0$

$$E(X) = E_0[X + RV2(1 - X)] \quad (\text{Eq. 5-9})$$

$$\rho(X) = \rho_0[X + RV3(1 - X)] \quad (\text{Eq. 5-10})$$

This case corresponds to the Voigt mixing rule (isostrain).

Topology Optimization

The input data to specify these relationships is:

1	2	3	4	5	6	7	8	9	10
TPROP	ID	PTYPE	PID	INIT					
+	"RULE"	RTYPE	RV1	RV2	RV3				

The RTYPE keyword could be POWER or MIX. POWER is used to create the power relationship and MIX is used to create the mix relationship. The default rule type is POWER. The POWER rule uses RV1 and RV2 and ignores RV3.

Progressive Rule

The value of RV1 is normally constant, however the use of a varying RV1 can be turned on with the **DOPT** parameter **TCYCLEM**. The **TCYCLE** entry defines a schedule of RV1 changes as the design cycles progress, allowing for the use of a progressive power rule.

5.2.4 Move Limits in Topology Optimization

As in size and shape optimization, *GENESIS* topology optimization uses an approximate problem to solve the optimization problem more efficiently. In general, the approximations are accurate close to the design variable value at which they were constructed. This makes it necessary to limit the range of design variable values for which the approximations are used. These limits will be used as temporary bounds in a particular design cycle. At the end of the optimization the move limits should not have affected the results.

The data to specify these move limits are: DELT and DTMIN. DELT is the fractional move limit. DTMIN is the minimum move limit. Their defaults are 1.0E-6 and 0.2 respectively. These defaults can be changed using the DOPT bulk data entry. Also, these values can be changed for an individual topology region using TPROP.

The input data to specify move limits is:

1	2	3	4	5	6	7	8	9	10
TPROP	ID	PTYPE	PID	INIT	TMIN	DELT	DTMIN		

The temporary bounds created using move limits at design cycle i for design variable X are:

$$LB_i = \text{MAX}[TMIN, X_i - \text{MAX}(DELT * X_i, DTMIN)] \quad (\text{Eq. 5-11})$$

$$UB_i = \text{MIN}[1.0, X_i + \text{MAX}(DELT * X_i, DTMIN)] \quad (\text{Eq. 5-12})$$

5.2.5 Fabrication Constraints

Fabrication constraints are used to enforce manufacturing requirements such as symmetry, extrusion and draw direction.

The data to specify these fabrication constraints are especified on TSYM1, TSYM2 or TSYM3 data entries. TSYMi entries are referenced on field 9 by referencing the TSYM identification number TSYMID:

1	2	3	4	5	6	7	8	9	10
TPROP	ID	PTYPE	PID	INIT	TMIN	DELT	DTMIN	TSYMID	

For more detatil on defining symmetry planes see [Defining the Symmetry Planes for Fabrication Constraints](#) (p. 378).

5.2.6 Extended Topology Regions

Typically a topology region references one property and when multiple properties are to be topologically designed then multiple TPROP entries are created. This case works well for most situations, but occasionally we would like to make one topology region reference multiple property IDs so that fabrication constraints can be enforced across multiple properties.

To handle this situation, the TPROP entry allows for optional continuation lines to list all additional properties

The format of TPROP with multiple properties is:

1	2	3	4	5	6	7	8	9	10
TPROP	ID	PTYPE	PID	INIT	TMIN	DELT	DTMIN	TSYMID	
+	"PLINK"	PID1	PID2	PID3	PID4	PID5	PID6	PID7	PID8
+		PID9	PID10	-etc.-					

Master Property

The property which is listed in the first line, in the field 4 of the TPROP entry is called master property.

Slave Properties

The properties that are listed in the PLINK list are called slave properties.

Slave properties can only have one master property. Slave properties have to be of the same type as the master property (e.g., a PBAR master can not have a PBEAM as a slave property).

5.2.7 Cloned Topology Regions

The previous section ([Extended Topology Regions](#) (p. 359)) described the process of combining multiple property IDs in a single topology region for the purpose of enforcing fabrication constraints across the whole collection of properties. There is another way to extend a topology region across multiple property IDs, such that the same final topology result is copied onto each property. This process is called *cloning*, and is specified using the **TPROP** input data.

TPROP data can be used to:

1. Define one or more regions as clones of an original design region
2. Define clones of parts that are symmetric by themselves
3. Define clones of parts that have fabrication constraints
4. Define clones that are translated and/or rotated copies of an original
5. Defined clones that are scaled versions of an original

The basic format for TPROP is:

1	2	3	4	5	6	7	8	9	10
TPROP	ID	PTYPEC	PIDC	CIDC	PTYPEP	PIDP	CIDP		

Example:

1	2	3	4	5	6	7	8	9	10
TPROP	12	PSHELL	101	2	PSHELL	100	1		

The parent property ID (PIDP) must be designed using **TPROP**.

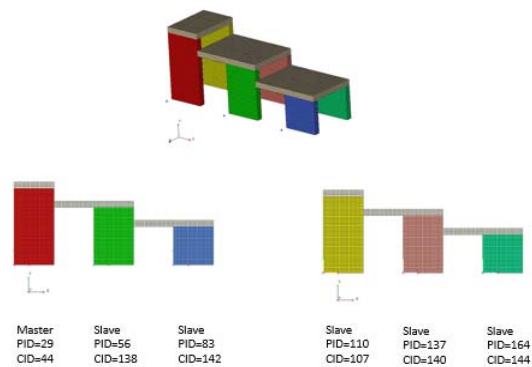
Cloning works as follows: Imagine translating and rotating the parent property and its coordinate system such that the parent coordinate system (CIDP) is aligned with the clone coordinate system (CIDC). Any place in the clone that overlaps this shifted parent phantom is forced to have the same topology value as the corresponding place in the parent. Parts of the clone property that do not overlap are designed independently, although they do inherit any fabrication constraints applied to the parent (shifted to the parent phantom).

The full format for TPROP is:

1	2	3	4	5	6	7	8	9	10
TPROP	ID	PTYPEC	PIDC	CIDC	PTYPEP	PIDP	CIDP		
+	SCALEX	SCALEY	SCALEZ						

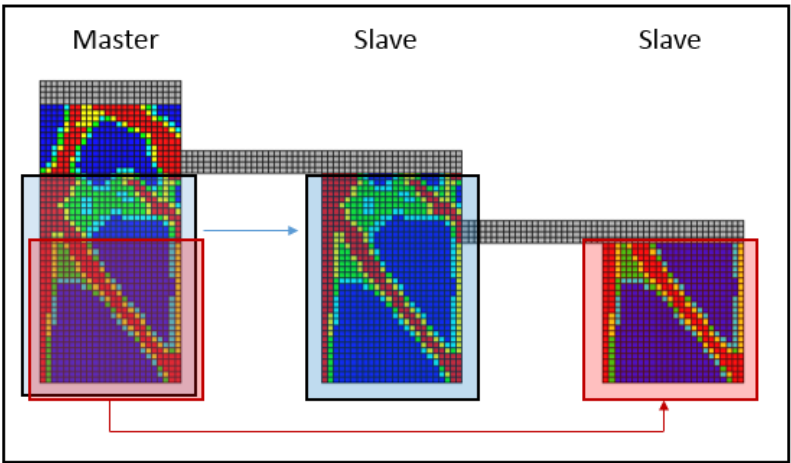
If any scale factors are different from 1.0 (the default), then after the parent phantom is shifted, it is scaled along each axis by the corresponding scale factor. Then the clone property is checked for overlapping regions. A negative scale factor along any axis will cause the parent phantom to “mirror flip” along that axis. This could be used to force mirror symmetry between a parent and a clone.

Example 1: Simple cloning: Extended region has 1 Master and 5 Slaves. The master is PSOLID 29. The slaves are PSOLID 110, 56, 137, 83 and 164.



	1	2	3	4	5	6	7	8	9	10
TPROP	1	PSOLID	29	0.3					1	
TSYM3	1	44								
+										
+	1.25	0.625								
TPROPC	2	PSOLID	110	107	PSOLID	29	44			
TPROPC	3	PSOLID	56	138	PSOLID	29	44			
TPROPC	4	PSOLID	137	140	PSOLID	29	44			
TPROPC	5	PSOLID	83	142	PSOLID	29	44			
TPROPC	6	PSOLID	164	144	PSOLID	29	44			

The following figure shows the results using cloning with no scaling.

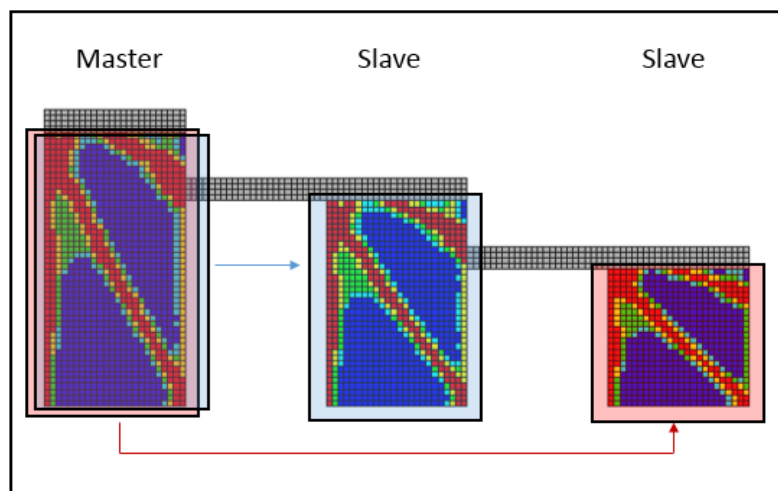


Topology Optimization

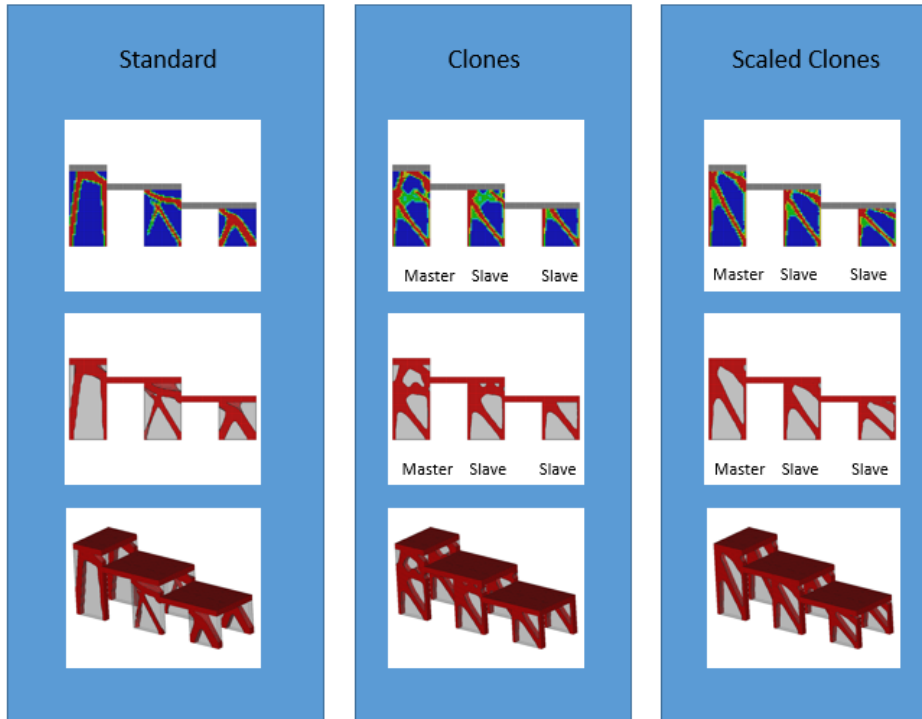
Example 2: Cloning and Scaling. The extended region has 1 Master and 5 Slaves. The master is PSOLID 29. Slaves are PSOLID 110, 56, 137, 83 and 164. PSOLID 110 is not scaled. PSSOLD 56 and 137 are sclaed so that their height is 75% of the height of their master (SCALEY=0.75). PSOLID 83 and 164 are scaled so that their height is 50% of their master (SCALEY=0.5).

	1	2	3	4	5	6	7	8	9	10
TPROP	1	PSOLID	29	0.3					1	
TSYM3	1	44								
+										
+	1.25	0.625								
TPROPC	2	PSOLID	110	107	PSOLID	29	44			
TPROPC	2	PSOLID	110	107	PSOLID	29	44			
TPROPC	3	PSOLID	56	138	PSOLID	29	44			
+	1.0	0.75	1.0							
TPROPC	4	PSOLID	137	140	PSOLID	29	44			
+	1.0	0.75	1.0							
TPROPC	5	PSOLID	83	142	PSOLID	29	44			
+	1.0	0.5	1.0							
TPROPC	6	PSOLID	164	144	PSOLID	29	44			
+	1.0	0.5	1.0							

The following figure shows the results using cloning and scaling.



The following figure shows results comparing three cases. The first case in the first column shows topology optimization results without using cloning. The second case in the second column shows results when clones are used without scaling. The third case in the third column shows results using cloning and scaling.



Notes:

The size of the master or slave parts does not matter. The slaves can be either smaller, or similar or larger than the masters.

The relative location of the coordinate systems of the master and the slaves defines the relative location of the cloned areas.

If a slave is hypothetically moved by the relative location and it does not coincide with the master, then no element in the slave region will be cloned.

Elements associated to the slave properties that do not have corresponding element in the master are still designed. These elements are designed using the same fabrication constraints as the parent.

The minimum and maximum member size of the slaves are scaled according the the scale factors specified by the user.

Topology Optimization

When minimum member size **is used** (defined in TSYM1, or TSYM2 or TSYM3 that is reference in the TPROP associated to the master), the meshes of the slaves (clones) do not need to have identical (in size and/or in shape) elements that match its master. In fact, a mesh assembled with triangular elements could be cloned to a mesh assembled with quadrilateral elements. On the other hand, if minimum member size **is not used** the slave(cloned) mesh needs to have elements that match its master. In this case, if not element in the slave mesh matches the element of its parent then no cloning occurs and the elements are designed independently.

When scaling is used and non member size is used, the mesh of the slaves have to be scaled version of the master mesh. To avoid this limitation, when using scalings on cloning, minimum member sized should be used.

Symmetric cloning:

To get clones that are symmetric copies of its parent, negative scales factors are typically need. In other words, SCALEX or SCALEY or SCALEZ typically needs to be -1.0 for symmetric cloning.

5.2.8 Selecting Responses

GENESIS can select any of the following structural analysis responses to be used in topology optimization:

Response Type (RTYPE)	Description
MASSFR	Fractional mass associated with the referenced property or total fractional mass
DISP	Displacement
SPCF	Reaction force
FREQ	Frequency
SENERGY	Strain energy associated with the sum of the strain energies referenced properties or the total strain energy
VMINDEX	Global von Mises stress index
TEMP	Temperature
HTC	Heat transfer compliance.
RELDISP	Relative displacement
CDISP	Grid contact clearance
CPRESS	Grid contact pressure or Grid glue connection pressure
INERTIA	System inertia
LAMA	Buckling eigenvalue
DDISP,	Dynamic displacement calculated by direct frequency response
DDISPS,	Shifted dynamic displacement calculated by direct frequency response
DVELO	Dynamic velocity calculated by direct frequency response
DVELOS	Shifted dynamic velocity calculated by direct frequency response
DACCE	Dynamic acceleration calculated by direct frequency response
DACCES	Shifted dynamic acceleration calculated by direct frequency response
MDISP	Dynamic displacement calculated by modal frequency response
MDISPS	Shifted dynamic displacement calculated by modal frequency response
MVELO	Dynamic velocity calculated by modal frequency response
MVELOS	Shifted dynamic velocity calculated by modal frequency response
MACCE	Dynamic acceleration calculated by modal frequency response

MACCES	Shifted dynamic acceleration calculated by modal frequency response
RMSDISP	Random root mean square dynamic displacement calculated by modal or direct frequency response
RMSVELO	Random root mean square dynamic velocity calculated by modal or direct frequency response
RMSACCE	Random root mean square dynamic acceleration calculated by modal or direct frequency response
RMSSTR	Random root mean square dynamic stress calculated by modal or direct frequency response
PSDDISP	Random power spectral density dynamic displacement calculated by modal or direct frequency response
PSDVELO	Random power spectral density dynamic velocity calculated by modal or direct frequency response
PSDACCE	Random power spectral density dynamic acceleration calculated by modal or direct frequency response
PSDDS	Shifted random power spectral density dynamic displacement calculated by modal or direct frequency response
PSDVS	Shifted random power spectral density dynamic velocity calculated by modal or direct frequency response
PSDAS	Shifted random power spectral density dynamic acceleration calculated by modal or direct frequency response
PSDSTR	Random power spectral density dynamic stress calculated by modal or direct frequency response
UFDISP	User function of dynamic displacement calculated by modal or direct frequency response
UFVELO	User function of dynamic displacement calculated by modal or direct frequency response
UFACCE	User function of dynamic acceleration calculated by modal or direct frequency response
UFDISPS	Shifted user function of dynamic displacement calculated by modal or direct frequency response
UFVELOS	Shifted user function of dynamic displacement calculated by modal or direct frequency response
UFACCES	Shifted user function of dynamic acceleration calculated by modal or direct frequency response
ERP	Equivalent radiated power
ERPS	Shifted equivalent radiated power

The TRESP1 data is used to tag the desired responses from analysis for the topology design model.

The format for the TRESP1 data is:

1	2	3	4	5	6	7	8	9	10
TRESP1	ID	LABEL	RTYPE	PTYPE		ATTA	ATT1		

Each TRESP1 has a required unique ID and an optional LABEL that can be used to identify it in the program output. The response type (RTYPE) must be MASSFR, DISP, SENERGY, FREQ, etc.

If the response is MASSFR and the PTYPE is a property type or PROP then the ATT1 is a property id. If the PTYPE is blank then the rest of the fields are not used. The mass fraction is the mass divided by the mass calculated if all design variables are 1.0.

If the response is SENERGY then the ATTi fields are optionally used. When one or more property IDs are listed this response is calculated by adding the strain energy of each element associated to all the listed properties. ATTA is not used for SENERGY.

If the response is DISP or RELDISP or SPCF or CDISP or CPRESS then PTYPE field is not used, the ATTA corresponds to component number and the ATT1 corresponds to a grid ID.

In addition, responses selected by **DRESP1** may also be influenced by the topology regions, and therefore any responses selectable on DRESP1 can effectively also be used in topology optimization. Care should be taken with DRESP1 stress responses on topology designed regions, as the stresses calculated with an intermediate Young's modulus may not be realistic.

The RTYPE, PTYPE, ATTA and ATTi data are summarized in the table below

RESPONSE	RTYPE	PTYPE	ATTA	ATTi
Static Displacement	DISP	Blank	Displacement component code (1-7).	Grid or Spoint ID, "THRU" or "ALL"
Static Relative Displacement	RELDISP	Blank	Displacement component code (1-7).	Grid or Spoint ID
Reaction Force	SPCF	Blank	Reaction Force component code (1-7).	Grid or Spoint ID
Grid Contact Pressure	CPRESS	Blank	Cpress component code (1).	Grid ID, "THRU" or "ALL"
Natural Frequency	FREQ	Mode number	Blank	Blank
Mass Fraction	MASSFR	Blank or PROP	Blank	Blank or Property ID
Static Strain Energy	SENERGY	Blank or PROP or PAXIS or PBAR or PBARL or PBEAM or PBEAML or PBUSH or PROD or PSHELL or PSHEAR or PCOMP or PCOMPG or PSOLID or PELAS or PVECTOR or PBUSH or PBUSHT or PGARP or PWELD or PK2UU	Blank	Blank or Property ID
Von mises index stress	VMINDEX	Blank	Blank	Blank
Temperature	TEMP	Blank	Blank	Grid ID, "THRU" or "ALL"
Heat Transfer Compliance	HTC	Blank	Blank	Blank
System Inertia Matrix Component	INERTIA	Blank	Inertia item code (1-22)	Blank
Buckling Load Factor	LAMA	Mode number	Blank	Blank

Dynamic Displacement, Velocity, Acceleration (From Direct Loadcases)	DDISP, DVELO, DACCE	Blank	Component code (1-24)	Grid or Spoint ID
Shifted Dynamic Displacement, Velocity, Acceleration (From Direct Loadcases)	DDISPS, DVELO, DACCES	Blank	Component code;(1-6)	ATT1: ID. ATTi (i>1): Grid or Spoint ID
Dynamic Displacement, Velocity, Acceleration (From Modal Loadcases)	MDISP, MVELO, MACCE	Blank	Component code (1-24)	Grid or Spoint ID
Shifted Dynamic Displacement, Velocity, Acceleration (From Modal Loadcases)	MDISPS, MVELO, MACCES	Blank	Component code: (1-6)	ATT1: ID. ATTi (i>1): Grid or Spoint ID
Random Dynamic Displacement, Velocity, Acceleration (From Direct or Modal Loadcases)	RMSDISP, RMSVELO, RMSACCE	Blank	Component code (1-6)	Grid or Spoint ID
Random Power Spectral Density Dynamic Displacement, Velocity, Acceleration (From Direct or Modal Loadcases)	PSDDISP, PSDVELO, PSDACCE	Blank	Component code (1-6).	Grid or Spoint ID

Shifted Random Power Spectral Density Dynamic Displacement, Velocity, Acceleration (From Direct or Modal Loadcases)	PSDDS, PSDVS, PSDAS	Blank	Component code (1-6).	ATT1: ID. ATTi (i>1): Grid or Spoint ID
User Function of Dynamic Displacement, Velocity, Acceleration	UFDISP, UFVELO, UFACCE	Blank	Field Point item code 1: Magnitude 2: Phase 3: Real 4: Imaginary	Field Point ID (Defined in UFDATA file)
Shifted User Function of Dynamic Displacement, Velocity, Acceleration (From Direct or Modal Loadcases)	UFDISPS, UFVELOS, UFACCES	Blank	Component code 1: Magnitude	ATT1: ID. ATTi (i>1):Field Point ID. Field Point Defined in UFDATA file.
Equivalent Radiated Power	ERP	Blank	Item code 1: Standard scale 2: Decibel Scale	Element set ID (Specified in ERPPNL entry)
Shifted Equivalent Radiated Power	ERPS	Blank	Component code 1: Standard scale 2: Decibel Scale	ATT1: DSHIFT ID. Element set ID (Specified in ERPPNL entry)

The following examples show the preparation of TRESP1 input data for different response types:

To tag the system mass fraction (mass fraction of all designed elements) the input data is:

1	2	3	4	5	6	7	8	9	10
TRESP1	21		MASSF R						

Note that the TRESP1 ID is 21, the optional label is not used, and that the PTYPE, ATTA and ATT1 are blank. To tag the mass associated with property number 45, the data entry is:

1	2	3	4	5	6	7	8	9	10
TRESP1	22		MASSF R	PROP			45		

To tag the u displacement of grid 78 the data entry is:

1	2	3	4	5	6	7	8	9	10
TRESP1	23	U_78	DISP			1	78		

Note that the PTYPE is left blank.

To tag the third system frequency the data entry is:

1	2	3	4	5	6	7	8	9	10
TRESP1	24	MODE3	FREQ	3					

Note that the LABEL is MODE3 and that the ATTA and ATT1 data are left blank.

To tag the system strain energy the data entry is:

1	2	3	4	5	6	7	8	9	10
TRESP1	25	ENERG Y	SENERGY						

Note that the LABEL is SENERGY and that the ATTA data is left blank.

5.2.9 Topology Design Objective Function

There are two types of data, TOBJ and TINDEX, to select the objective function. TOBJ is to select a single response and TINDEX is to construct an objective function that is a linear combination of responses or their reciprocals. Only one of TOBJ or TINDEX may appear in the input data.

The TOBJ data is used to flag one of the TRESP1 responses as the topology design objective function. The format of the TOBJ input data is:

1	2	3	4	5	6	7	8	9	10
TOBJ	RID	LABEL	LID	MIN/MAX					

The RID corresponds to the TRESP1 ID that is to become the objective function response. The optional LABEL is used to help to identify the objective function in the program output. The LID corresponds to the LOADCASE number from which the response will be calculated (default is the first LOADCASE). The load case specified by LID must be a static or eigenvalue load case. Finally, the last data specifies whether the objective function is to be minimized or maximized (default is minimized). If the RID corresponds to MASSFR then the LID is ignored.

For example, the TOBJ data for strain energy minimization could be:

1	2	3	4	5	6	7	8	9	10
TOBJ	10	MIN_SE	100						
TRESP1	10	ENERGY	SENERGY						

Here 10 is the TRESP1 ID that tags the system strain energy. The 100 indicates that the strain energy calculated in load case 100 is to be used.

The objective function to maximize the first natural frequency could be:

1	2	3	4	5	6	7	8	9	10
TOBJ	20	MAX_FRQ	200	MAX					
TRESP1	20	MODE1	FREQ	1					

Here 20 is the TRESP1 ID that tags the frequency. The 200 indicates that the frequency calculated in load case 200 is to be used.

The TINDEX data is used to specify the responses the user want to be included in a compliance index objective function. This is done by specifying TRESP IDs, weighting factors and LOADCASE numbers in the format:

1	2	3	4	5	6	7	8	9	10
TINDEX	RID1	LID1	W1	RID2	LID2	W2			
+	RID3	LID3	W3	-etc.-					

RID_i is the response number, LID_i is the load case number and W_i is the weighting factor.

TINDEX is an alternative way to select a topology objective function. The compliance index function is a weighted sum of response terms. How each term enters into the index function depends on the DOPT parameter, TINDEXM.

If TINDEXM is 0 or 1, then responses are normalized by their values in the first design cycle. If TINDEXM is 2 or 3, then responses are not normalized.

- TINDEXM = 0 or 1

$$R_i = \frac{\text{Response}_i}{|\text{Response}_{0i}|} \quad (\text{Eq. 5-13})$$

- TINDEXM = 2 or 3

$$R_i = \text{Response}_i \quad (\text{Eq. 5-14})$$

The compliance index objective function is calculated using the following equation:

$$T = \sum_{i=1} f_i \quad (\text{Eq. 5-15})$$

If the DOPT parameter TINDEXM is 0 or 2, then the compliance index objective function terms are calculated using reciprocals for negative weighting factors. If TINDEXM is 1 or 3, then the compliance index objective function is a straight summation.

- TINDEXM = 0 or 2

$$f_i = \begin{cases} W_i \cdot R_i & \text{if } W_i > 0 \\ \frac{-W_i}{R_i} & \text{if } W_i < 0 \end{cases} \quad (\text{Eq. 5-16})$$

- TINDEXM = 1 or 3

$$f_i = W_i R_i \quad (\text{Eq. 5-17})$$

Topology Optimization

If the weighting factor, W_i , is the word MULT or DIV, then the corresponding response, R_i , does not make a new term f_i in the compliance index function. Instead, R_i multiplies or divides the immediately previous term f_j ($j < i$) in the compliance index function for which a real weighting factor was given. For example:

	1	2	3	4	5	6	7	8	9	10
TINDEX	1	101	1.0							
+	2	102	MULT							
+	3	103	2.0							
+	4	104	DIV							

For the above entry, the objective function is $T = R_1 R_2 + 2.0 \frac{R_3}{R_4}$

Typically, the compliance index relationship is used for combining strain energies and/or frequency responses. In this case typically the weighting factor used are positive for the strain energy responses and negative for frequency responses.

The reason for this is that *GENESIS* will minimize the compliance index function. The choice of a positive weighting factor for strain energy and negative for frequency will make the structure stiffer.

The TINDEX function also accepts the use of MASSFR or DISP responses.

In any case, it is up to the user to decide the sign of the weighting factors.

Examples:

Minimize

$$TI = \frac{1.0 \times (SEnergy) \text{ (load case 10)}}{SEnergy \text{ (load case 10 initial)}} + \frac{100 \times Freq \text{ (mode 1, load case 20 initial)}}{Freq \text{ (mode 1, load case 20)}}$$

The data for this problem could be:

	1	2	3	4	5	6	7	8	9	10
TRESP1	1	ENERGY	SENERGY							
TRESP1	2	MODE1	FREQ	1						
\$										
TINDEX	1	10	1.0							
+	2	20	-100.0							
DOPT										
+	TINDEXM	0								

Since the default value of the TINDEXM parameter is 0, the DOPT data statement is not required.

Instead of using the reciprocal of the frequencies in the compliance index function the user could use the negative components as it is shown in the next problem:

$$TI = \frac{1.0 \times (SEnergy) \text{ (load case 10)}}{SEnergy \text{ (load case 10 initial)}} - \frac{100 \times Freq \text{ (mode 1, load case 20)}}{Freq \text{ (mode 1, load case 20 initial)}}$$

The data for this problem could be:

	1	2	3	4	5	6	7	8	9	10
TRESP1	1	ENERGY	SENERGY							
TRESP1	2	MODE1	FREQ	1						
\$										
TINDEX	1	10	1.0							
+	2	20	-100.0							
DOPT										
+	TINDEXM	1								

In this case the DOPT parameter TINDEXM is required to override the default value of 0.

$$\text{Minimize TI} = 1.0 * \text{Strain Energy (load case 10)} + \frac{100.0}{\text{Freq (mode 1, load case 20)}}$$

The data for this problem could be:

	1	2	3	4	5	6	7	8	9	10
TRESP1	1	ENERGY	SENERGY							
TRESP1	2	MODE1	FREQ	1						
\$										
TINDEX	1	10	1.0							
+	2	20	-100.0							
DOPT										
+	TINDEX M	2								

Instead of using the reciprocal of the frequencies in the compliance index function the user could use the negative components as it is shown in the next problem:

$$\text{TI} = 1.0 * \text{Strain Energy (load case 10)} - 100 * \text{Freq (mode 1, load case 20)}$$

The data for this problem could be:

	1	2	3	4	5	6	7	8	9	10
TRESP1	1	ENERGY	SENERGY							
TRESP1	2	MODE1	FREQ	1						
\$										
TINDEX	1	10	1.0							
+	2	20	-100.0							
DOPT										
+	TINDEXM	3								

In this case the DOPT parameter TINDEXM is required to override the default value of 0.

5.2.10 Topology Constraints

The TCONS data is used to place lower and upper bound constraints on the TRESP1, TRESP2 or TRESP3 responses. This is done by specifying the TRESPx ID, LOADCASE number, and lower and upper bound values in the format:

1	2	3	4	5	6	7	8	9	10
TCONS	RID	LID1	LB1	UB1	LID2	LB2	UB2		
+		LID3	LB3	UB3	...				

Here LID_i is the load case number, LB_i and UB_i are respectively the lower and upper bounds for load case *i*.

Note that different bounds can be used in different load cases and that the responses do not have to be constrained in all load cases. If the response has the same bound in all the load cases then the following alternate format of TCONS can be used:

1	2	3	4	5	6	7	8	9	10
TCONS	RID	"ALL"	LB1	UB1					

If this format is used in input data that contains static and frequency load cases, *GENESIS* will only constrain the responses in the appropriate load cases (static responses in static load cases, frequency responses in frequency calculation load cases).

If the RID corresponds to MASSFR then the LID is ignored.

For FREQ responses only one bound is allowed per TCONS data. If the two bounds are needed simply use two TRESP1-TCONS sets.

Constraints can also be defined using the TCONS2 entry. The purpose of TCONS2 is to use constraint bounds relative to a response's initial value. The format of TCONS2 is similar to TCONS:

1	2	3	4	5	6	7	8	9	10
TCONS2	RID	LID1	LBF1	UBF1	LID2	LBF2	UBF2		
+		LID3	LBF3	UBF3	...				

where, LBF_i is a factor that scales the initial value of the response to calculate the actual lower bound and UBF_i is a factor that scales the initial value of the response to calculate the actual upper bound.

5.2.11 Defining the Symmetry Planes for Fabrication Constraints

Each topology region can be designed using a different set of symmetry planes and fabrication constraints. Symmetry planes and the fabrication constraints can be defined using TSYM1, TSYM2 or TSYM3 entries. These entries must to be referenced by TPROP entries to be used.

Defining the Symmetry Planes

The TSYM1 entry is used to create a fabrication system using 3 grids. The basic format used in TSYM1 is:

1	2	3	4	5	6	7	8	9	10
TSYM1	ID	G1	G2	G3					

The ID is the TSYM identification number. G1, G2 and G3 are the grid identification numbers of three non-colinear grids. The XZ plane is the plane containing G1,G2,G3 (as in the XZ plane of a CORD1R coordinate system created with G1, G2, G3). The XY plane is the plane that contains G1 and is normal to the line defined by G1 and G2. The YZ plane is the plane that is perpendicular to the ZX and XY planes and contains the grid G1.

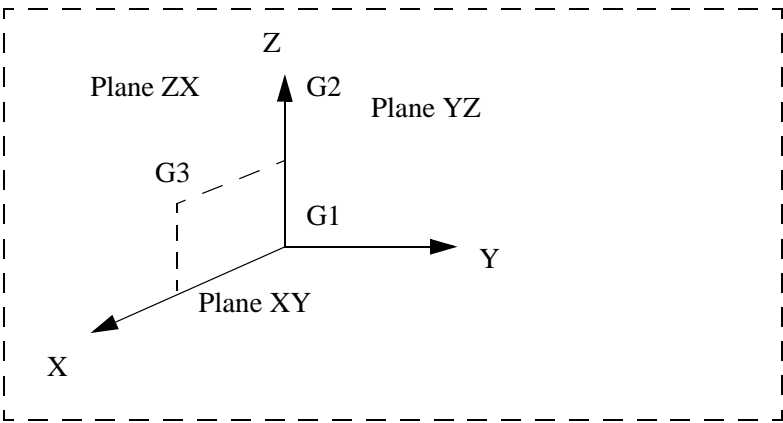


Figure 5-1

The TSYM2 entry is used to create a fabrication system using 3 points. The basic format used in TSYM2 is:

1	2	3	4	5	6	7	8	9	10
TSYM2	ID	CID	A1	A2	A3	B1	B2	B3	
+	C1	C2	C3						

The ID is the TSYM identification number. (A1,A2,A3), (B1,B2,B3) and (C1,C2,C3) are the coordinates of the 3 points written in the CID coordinate system. The three point must not be colinear.

The XZ plane is the plane containing (A1,A2,A3), (B1,B2,B3) and (C1,C2,C3) points (as in the XZ plane of a CORD2R coordinate system created with (A1,A2,A3), (B1,B2,B3), (C1,C2,C3). The XY plane is the plane that contains the point (A1,A2,A3) and is normal to the line defined by (A1,A2,A3) and (B1,B2,B3). The YZ plane is the plane that is perpendicular to the ZX and XY planes and contains the point (A1,A2,A3).

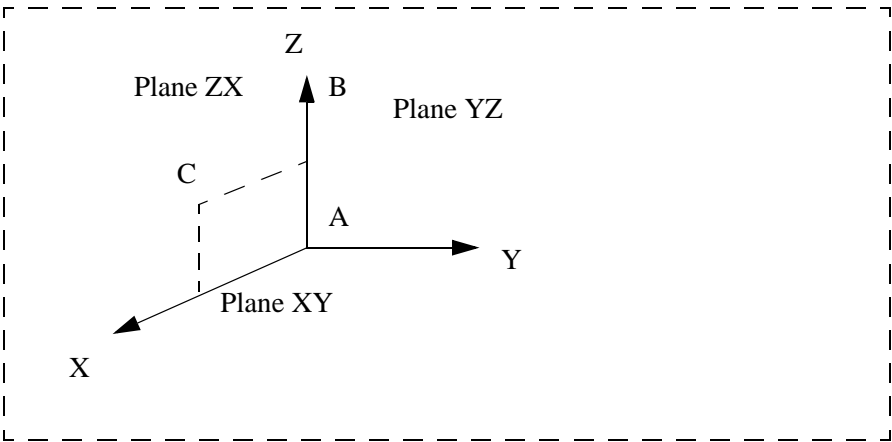


Figure 5-2

The TSYM3 data is used to set coordinate system as a fabrication system.The basic format used in TSYM3 is:

1	2	3	4	5	6	7	8	9	10
TSYM3	ID	CID							

The ID is the TSYM identification number. The CID number corresponds to a coordinate system identification number.

The planes of symmetries are built using the axis of the selected coordinate system. In other words, The XY symmetry plane is the plane containing the X and Y axis of the coordinate system. The YZ symmetry plane is the plane containing the Y and Z axis of the coordinate system. Finally, the ZX symmetry plane is the plane containing the Z and X axis of the coordinate system.

5.2.12 Defining the Fabrication Constraints

The fabrication constraints are defined in TSYM1, TSYM2 or TSYM3 data. Sixteen varieties of fabrication constraints are currently available for each the three directions plus uniform XYZ make a total of 55 types of fabrication constraints:

TYPE	Description of Fabrication Constraints
MXY	Mirror symmetry with respect to the XY plane
MYZ	Mirror symmetry with respect to the YZ pane
MZX	Mirror symmetry with respect to the ZX plane
CX	Cyclic symmetry about the X axis
CY	Cyclic symmetry about the Y axis
CZ	Cyclic symmetry about the Z axis
EX	Extrusion along the X axis
EY	Extrusion along the Y axis
EZ	Extrusion along the Z axis
PX	Periodic pattern repetition in the X direction (with user specified pitch pattern distance DISTX)
PY	Periodic pattern repetition in the Y direction (with user specified pitch pattern distance DISTY)
PZ	Periodic pattern repetition in the Z direction (with user specified pitch pattern distance DISTZ)
P0X	Symmetric Periodic pattern repetition in the X and -X directions (with user specified pitch pattern distance DISTX)
P0Y	Symmetric Periodic pattern repetition in the Y and -Y directions (with user specified pitch pattern distance DISTY)
P0Z	Symmetric Periodic pattern repetition in the Z and -Z directions (with user specified pitch pattern distance DISTZ)
FBX	Filling in the +X direction
FBY	Filling in the +Y direction
FBZ	Filling in the +Z direction
FTX	Filling in the -X direction

FTY	Filling in the -Y direction
FTZ	Filling in the -Z direction
FSX	Filling simulatneously from +X and -X directions (fill from the outside in)
FSY	Filling simulatneously from +Y and -Y directions (fill from the outside in)
FSZ	Filling simulatneously from +Z and -Z directions (fill from the outside in)
FGX	Filling in +X and -X directions from general surface (fill from the inside out)
FGY	Filling in +Y and -Y directions from general surface (fill from the inside out)
FGZ	Filling in +Z and -Z directions from general surface (fill from the inside out)
F0X	Filling symmetrically from X=0.0 toward the top and bottom (filling from the inside out)
F0Y	Filling symmetrically from Y=0.0 toward the top and bottom (filling from the inside out)
F0Z	Filling symmetrically from Z=0.0 toward the top and bottom (filling from the inside out)
RBX	Radial Filling From the Inner Surface about the X axis
RBY	Radial Filling From the Inner Surface about the Y axis
RBZ	Radial Filling From the Inner Surface about the Z axis
RTX	Radial Filling From the Outer Surface about the X axis
RTY	Radial Filling From the Outer Surface about the Y axis
RTZ	Radial Filling From the Outer Surface about the Z axis
RGX	Radial Filling From the General Surface about the X axis (from the interior to the inner and outer surfaces)
RGY	Radial Filling From the General Surface about the Y axis (from the interior to the inner and outer surfaces)
RGZ	Radial Filling From the General Surface about the Z axis (from the interior to the inner and outer surfaces)
KX	Radial Spokes about the X axis

KY	Radial Spokes about the Y axis
KZ	Radial Spokes about the Z axis
SBX	Sheet forming normal to +X (1 Layer) (initial configuration starts at the bottom)
SBY	Sheet forming normal to +Y (1 Layer) (initial configuration starts at the bottom)
SBZ	Sheet forming normal to +Z (1 Layer) (initial configuration starts at the bottom)
STX	Sheet forming normal to +X (1 Layer) (initial configuration starts at the top)
STY	Sheet forming normal to +Y (1 Layer) (initial configuration starts at the top)
STZ	Sheet forming normal to +Z (1 Layer) (initial configuration starts at the top)
S2X	Sheet forming normal to +X (2 Layer) (initial configuration of first layer starts at the top; initial configuration of second layer starts at the bottom)
S2Y	Sheet forming normal to +Y (2 Layer) (initial configuration of first layer starts at the top; initial configuration of second layer starts at the bottom)
S2Z	Sheet forming normal to +Z (2 Layer) (initial configuration of first layer starts at the top; initial configuration of second layer starts at the bottom)
UXY	Uniform in planes parallel to the XY plane
UYZ	Uniform in planes parallel to the YZ pane
UZX	Uniform in planes parallel to the ZX plane
UXYZ	Uniform value for all elements

The format of TSYM1 with fabrication constraints is:

1	2	3	4	5	6	7	8	9	10
TSYM1	ID	G1	G2	G3	n	NSECT	DISTX	DISTY	DISTZ
+	TYPE1	TYPE2	TYPE3						
+	SYMV1	SYMV2							
+	STHICK	PUNCH	VOIDDNS	OFFSET					

The format of TSYM2 with fabrication constraints is:

1	2	3	4	5	6	7	8	9	10
TSYM2	ID	RID	A1	A2	A3	B1	B2	B3	
+	C1	C2	C3	n	NSECT	DISTX	DISTY	DISTZ	
+	TYPE1	TYPE2	TYPE3						
+	SYMV1	SYMV2	SYMV3	SYMV4					
+	STHICK	PUNCH	VOIDDNS	OFFSET					

The format of TSYM3 with fabrication constraints is:

1	2	3	4	5	6	7	8	9	10
TSYM3	ID	CID	n	NSECT	DISTX	DISTY	DISTZ		
+	TYPE1	TYPE2	TYPE3						
+	SYMV1	SYMV2							
+	STHICK	PUNCH	VOIDDNS	OFFSET					

Minimum Member Size

On TSYM1, TSYM2 and TSYM3, the SYMV1 and SYMV2 fields are used to specified the **spacing** and the **spread** of the design variables. Together, these control the **minimum member size**. The first quantity is used to obtain more buildable solutions while the second helps to smooth the topology results. SYMV1 is currently required for all filling and stamping constraints and is optional for extrusion constraints. Using a larger value for SYMV1 results in fewer design variables.

Maximum Member Size

The SYMV3 and SYMV4 fields are used to specified the **maximum member size** and the **minimum gap** of parts created by topology. SYMV3 and SYMV4 are used to reduce local mass concentration. SYMV4 requires the use of SYMV3. While SYMV3 requires the use of SYMV1. The default value of SYMV4 when SYMV3 is present is SYMV3. If SYMV3 is not used then maximum member size control is not used.

There are no requirements on the value of SYMV3, however if used, it should be larger than $SYMV1 + 2 * SYMV2$ to avoid grey answers. Typically, SYMV3 should be 3 times the size of the actual minimum member size ($SYMV1 + 2 * SYMV2$).

Stamping Constraints

STHICK, PUNCH, VOIDDNS and OFFSET are parameters to control the stamping constraints. STHICK is used to define the thickness of the sheets, PUNCH is to select between allowing through holes punched in the sheet (YES value) or not (NO value). VOIDDNS is the density of the voids (a small value such 0.001 is recommended instead of 0.0 to avoid singularities). OFFSET is a parameter to offset the initial location of the sheets.

No-Hole option on Casting Constraints

STHICK, PUNCH are parameters to control whether a cast design is allowed to have through-holes. PUNCH is to select between allowing through-holes (YES value) or not (NO value). If PUNCH is NO, STHICK is used to define the minimum thickness in the casting. These parameters can be used to create watertight designs. The default for PUNCH is YES.

Use of Mirror Symmetries

Up to three mirror symmetries are allowed per TSYMi data. The finite element mesh must have the mirror symmetry properties in order for the TSYMi mirror symmetries to be enforced. The following examples illustrate the use of 3 mirror symmetries.

Example: Triple symmetry conditions:

Impose triple symmetry on all elements referencing PID=101, 102, 103 and 104.

The topology data for this problem could be:

1	2	3	4	5	6	7	8	9	10
\$ Topology designable regions									
TPROP	1	101	POWER	0.3				201	
TPROP	2	102	POWER	0.3				201	
TPROP	3	103	POWER	0.3				201	
TPROP	4	104	POWER	0.3				201	
\$ Symmetry Plane and Fabrication constraints									
TSYM1	201	17	18	19					
+	MYZ	MZX	MXY						

Mirror symmetries can also be combined with other fabrication constraints.

Use of Cyclic Symmetry

The TSYM1, TSYM2 or TSYM3 data entries may be used to enforce cyclic symmetry in the topology model. The finite element mesh must have the cyclic symmetry property in order for the TSYMi cyclic symmetry to be enforced. The formats in this case require using the field that specifies the number of cyclic symmetries n:

1	2	3	4	5	6	7	8	9	10
TSYM1	ID	G1	G2	G3	n				

1	2	3	4	5	6	7	8	9	10
TSYM2	ID	RID	A1	A2	A3	B1	B2	B3	
+	C1	C2	C3	n					

1	2	3	4	5	6	7	8	9	10
TSYM3	ID	CID	n						

5

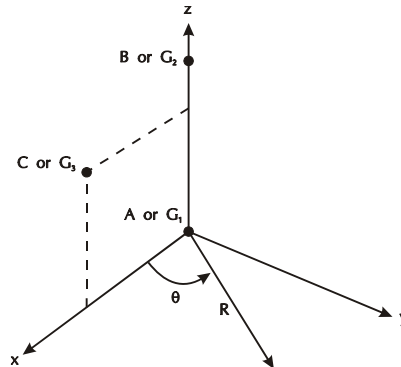
Example: Cyclic Symmetry about the z axis

Impose cyclic symmetry about the z direction on all elements referencing PID=71 forcing that the topology patterns are repeated every 60 degrees.

The topology data for this problem could be:

1	2	3	4	5	6	7	8	9	10
\$ Topology designable regions									
TPROP	1	71	POWER	0.3				255	
\$ Symmetry Plane, Fabrication constraint and Minimum Size									
TSYM1	255	7	18	15	6				
+	CZ								

The TSYM1 255 entry is used in this case defines the x,y,z directions and selects the cyclic symmetry constraint about the z axis (CZ).



The number $n=6$ ($360/60$) specifies the number of cyclically symmetric sections about the z-axis (G_1 - G_2) that should be used. The corresponding elements in each section of the following figure would be linked.

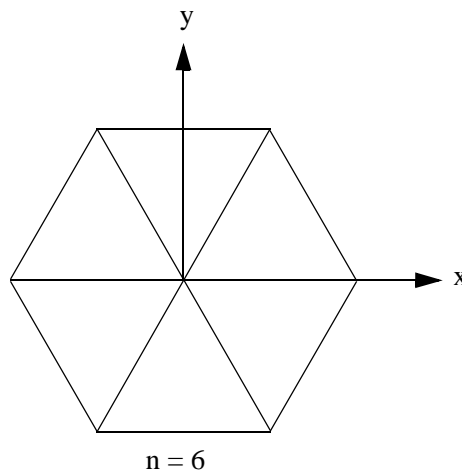


Figure 5-3

Note that the first symmetry section starts at the x-axis in the above figure.

Use of Extrusion Constraint

Extrusion constraints are used to impose extrusion requirements on a given topology region. To use this type of fabrication constraint, it is necessary to define a coordinate system and select the direction of extrusion.

Example: Extrusion along the z direction

Impose extrusion constraints on the z direction on all elements referencing PID=61, 62, 63, 64 and 65.

The topology data for this problem could be:

	1	2	3	4	5	6	7	8	9	10
\$ Topology designable regions										
TPROP	1	61	POWER	0.3					233	
TPROP	1	62	POWER	0.3					233	
TPROP	1	63	POWER	0.3					233	
TPROP	1	64	POWER	0.3					233	
TPROP	1	65	POWER	0.3					233	
\$ Symmetry Plane, Fabrication constraint and Minum Size										
TSYM1	233	7	18	15						
+	EZ									
+	5.0									

The TSYM1 233 entry used in this case defines the x,y,z directions and selects the extrusion fabrication constraint and its direction (EZ).

Extrusion constraints can be imposed by using one of two different methods. The first method requires a minimum member size to be specified. The second method requires that the finite element mesh contain similar elements along the extrusion direction (i.e., solid elements created by extruding a 2-D mesh). The first method is necessary for structures where the mesh is not extruded or when the minimum size is required. If the SYMV1 field on the TSYM_i entry is not blank, then the first method is used, otherwise the second method is used.

Use of Filling Constraint

Filling constraints are used to impose fabrication requirements, such as castability, where it is important that a part does not “lock the mold”. This constraint is imposed by requiring that material can only be added into the region by “filling up” in a given direction. To use this type of fabrication constraints it is necessary to define a coordinate system and select the filling direction (mold pull-off direction).

If the design needs to be watertight, then field PUNCH should be set to NO and STHICK should be set to the minimum allowable thickness.

Example: Filling in the +x direction

Impose filling constraints on the +x direction on all elements referencing PID=51, 52 and 53.

The topology data for this problem could be:

	1	2	3	4	5	6	7	8	9	10
\$ Topology designable regions										
TPROP	1	51	POWER	0.3					107	
TPROP	1	52	POWER	0.3					107	
TPROP	1	53	POWER	0.3					107	
\$ Symmetry Plane, Fabrication constraint and Minimum Size										
TSYM1	107	7	18	15						
+	FBX									
+	5.0									

The TSYM1 107 entry used in this case is to define the x,y,z directions, and to select the fabrication constraint (FBX).

To use filling constraints, the SYMV1 field on the TSYM_i entry must be given. In this case a minimum size of 5.0 is used.

Filling Types

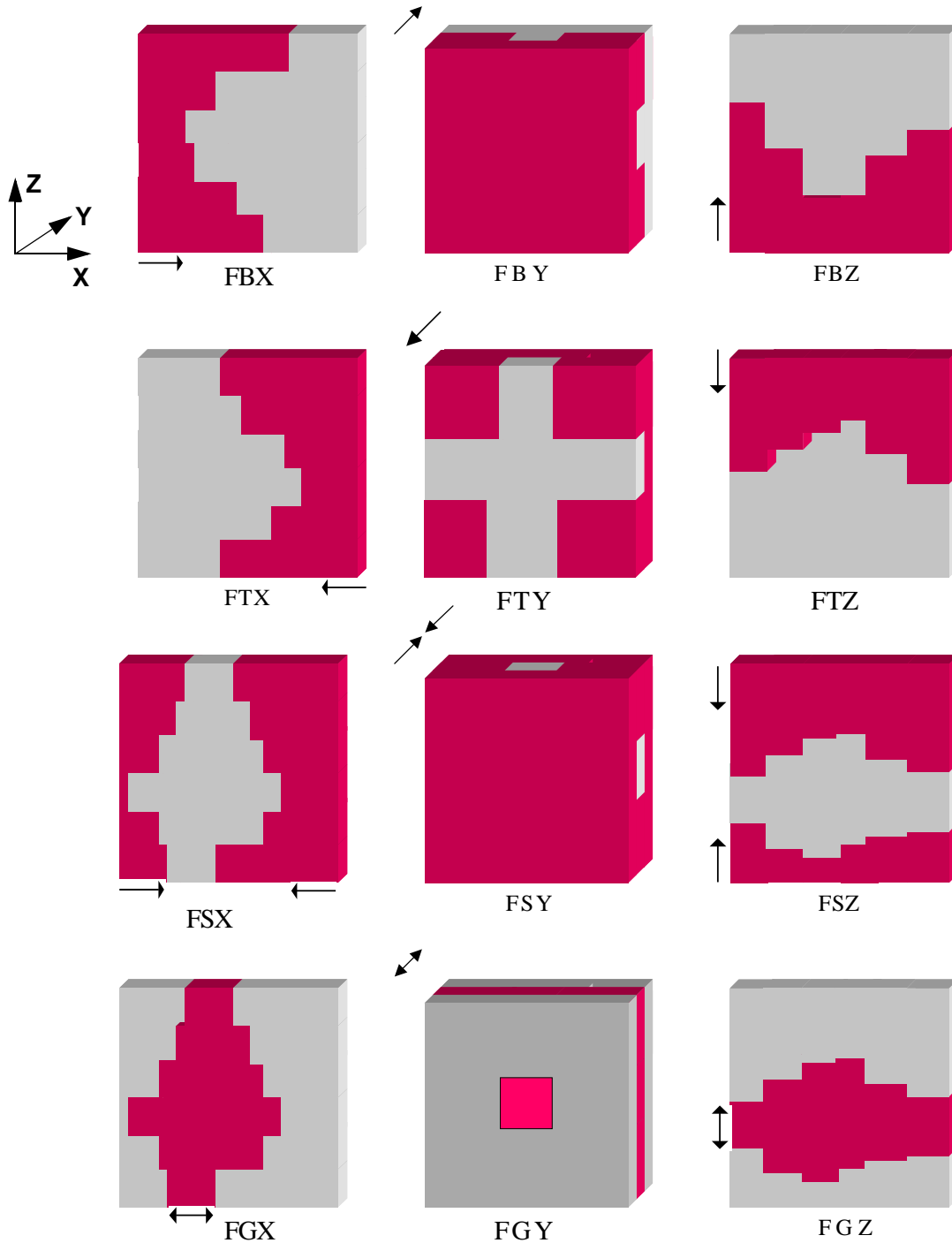


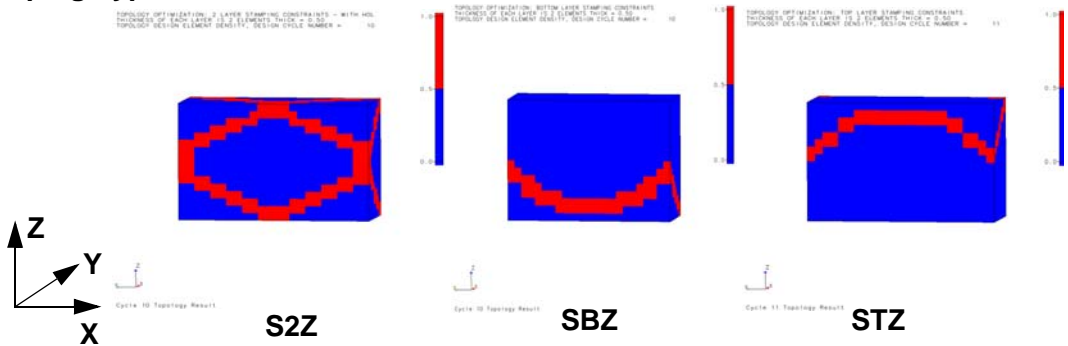
Figure 5-4

In red (dark) is the material to keep, in grey (light) are the voids.

Use of Sheet Forming Constraint

Sheet forming constraints are used to impose fabrication requirements, so that the final structure can be built using one or two stamped sheets. To use this type of fabrication constraints it is necessary to define a coordinate system and select the stamping direction. Holes punched through the sheets can be allowed or disallowed.

Stamping Types



In red (dark) is the material to keep, in blue are the voids.

Example: Two Sheets Stamped in the z-direction

The topology data for this problem could be:

	1	2	3	4	5	6	7	8	9	10
\$ Topology designable regions										
TPROP	1	51	POWER	0.3					101	
\$ Symmetry Plane, Fabrication constraint, Minimum Size and Sheet requirements										
TSYM1	109	7	18	15						
+	S2Z									
+	5.0									
+	2.0	NO								

The TSYM1 109 entry used in this case is to define the x,y,z directions, and to select the fabrication constraint (S2Z).

The refinement size is 5.0. In other words, the change of curvature in the sheet could be noticed on ranges of 5.0 units or more.

The desired thickness is 2.0. No punch holes are desired in the sheets.

Example: One Sheet Stamped in the z-direction

The topology data for this problem could be:

	1	2	3	4	5	6	7	8	9	10
\$ Topology designable regions										
TPROP	1	51	POWER	0.3					101	
\$ Symmetry Plane, Fabrication constraint, Minimum Size and Sheet requirements										
TSYM1	120	7	18	15						
+	SBZ									
+	5.0									
+	2.0	NO								

The TSYM1 120 entry used is in this case is to define the x,y,z directions, and to select the fabrication constraint (SBZ).

The refinement size is 5.0. In other words, the change of curvature in the sheet could be noticed on ranges of 5.0 units or more.

The desired thickness is 2.0. No punch holes are desired in the sheets.

5.2.13 Combining Fabrication Constraints

Sometimes it is necessary to impose multiple fabrication constraints on a given topology region simultaneously. Up to three fabrication constraints can be used per region. However, not all can be mixed together.

Here is a list of the possible combinations in a single topology region:

1. Two or three mirror symmetry constraints
2. One cyclic constraint can be mixed with one mirror symmetry as long as the axis of cyclic symmetry is normal to the plane of mirror symmetry.
3. One extrusion constraint can be mixed with one or two mirror symmetry constraints, as long as the extrusion direction is parallel to the plane(s) of mirror symmetry.
4. One extrusion constraint can be mixed with one cyclic symmetry constraint, as long as the extrusion direction and the cyclic axis are the same.
5. One filling or sheet forming constraint can be mixed with one or two mirror symmetry constraints, as long as the filling or stamping direction is parallel to the plane(s) of mirror symmetry.
6. One filling or sheet forming constraint can be mixed with one cyclic symmetry constraint, as long as the filling or stamping direction and the cyclic axis are the same.
7. One uniform constraint can be mixed with the mirror constraint of the same plane.
8. More precisely, here is the list of possible combinations (not including periodic):

$MXY+MYZ, MXY+MZX, MYZ+MZX, MXY+MYZ+MZX$
$CX+MYZ, CY+MZX, CZ+MXY$
$EX+MXY, EX+MZX, EX+MXY+MZX$ $EY+MYZ, EY+MXY, EY+MYZ+MXY$ $EZ+MZX, EZ+MYZ, EZ+MZX+MYZ$
$EX+CX, EY+CY, EZ+CZ$
$UYZ+MYZ, UZX+MZX, UXY+MXY$
$FiX+MXY, FiX+MZX, FiX+MXY+MZX$ $FiY+MYZ, FiY+MXY, FiY+MYZ+MXY$ $FiZ+MZX, FiZ+MYZ, FiZ+MZX+MYZ$
$FiX+CX, FiY+CY, FiZ+CZ$
$RiX+MXY, RiX+MYZ, RiX+ MZX,$ $RiX+MXY+MYZ, RiX+MXY+MZX, RiX+MYZ+MZX$ $RiY+MXY, RiY+MYZ, RiY+ MZX,$ $RiY+MXY+MYZ, RiY+MXY+MZX, RiY+MYZ+MZX$ $RiZ+MXY, RiZ+MYZ, RiZ+ MZX,$ $RiZ+MXY+MYZ, RiZ+MXY+MZX, RiZ+MYZ+MZX$
$RiX+CX, RiY+CY, RiZ+CZ$
$RiX+EX, RiY+EY, RiZ+EZ$

$RiX+CX+EX, RiY+CY+EY, RiZ+CZ+EZ$
$RiX+CX+MYZ, RiY+CY+MZX, RiZ+CZ+MXZ$
$RiX+EX+MXZ, RiX+EX+MZX, RiY+EY+MXZ, RiY+EY+MYZ, RiZ+EZ+MYZ, RiZ+EZ+MZX,$

In the above table:

FiX is one of $FBX, FTX, FSX, FGX, F0X, SBX, STX$ or $S2X$

FiY is one of $FBY, FTY, FSY, FGY, F0Y, SBY, STY$ or $S2Y$

FiZ is one of $FBZ, FTZ, FSZ, FGZ, F0Z, SBZ, STZ$ or $S2Z$

RiX is one of RBX, RTX, RGX , or KX

RiY is one of RBZ, RTZ, RGZ , or KZ

RiZ is one of RBZ, RTZ, RGZ , or KZ

- Periodic structures with repetitions on the x directions requires using PX (or $P0X$) and $DISTX$. Periodic structures with repetitions on the y directions requires using PY (or $P0Y$) and $DISTY$. Periodic structures with repetitions on the z directions requires using PZ (or $P0Z$) and $DISTZ$.
- Periodic fabrication constraints and their valid combinations with other fabrication constraints are shown next:

Types that can combined with Periodic	Combinations	Number of combinations
2 or 3 Periodic	$PX+PY, PX+PZ, PY+PZ, PX+PY+PZ$	4
Periodic + 1 or 2 Mirrors	$PX+MXZ, PX+MZX, PX+MXZ+MZX, PY+MXZ, PY+MYZ, PY+MXZ+MYZ, PZ+MXZ, PZ+MYZ, PZ+MXZ+MYZ$	9
2 Periodic + Mirror	$PX+PY+MXZ, PX+PZ+MZX, PY+PZ+MYZ$	3
Periodic + Cyclic	$PX+CX, PY+CY, PZ+CZ$	3
Periodic + Extrusion	$PX+EY, PX+EZ, PY+EX, PY+EZ, PZ+EX, PZ+EY$	6
2 Periodic + Extrusion	$PX+PY+EZ, PX+PZ+EY, PY+PZ+EX$	3
Periodic + Mirror + Extrusion	$PX+MXZ+EY, PX+MZX+EZ, PY+MXZ+EX, PY+MYZ+EZ, PZ+MXZ+EX, PZ+MYZ+EY$	6
Periodic + Uniform	$PX+UYZ, PY+UZX, PZ+UXY$	3

Periodic + Filling	PX+FiY,PX+FiZ, PY+FiX,PY+FiZ, PZ+FiX, PZ+FiY	6
2 Periodic + Filling	PX+PY+FiZ, PX+PZ+FiY, PY+PZ+FiX,	3
Periodic +Mirror + Filling	PX+MXY+FiY, PX+MZX+FiZ, PY+MXY+FiX,PY+MYZ+FiZ, PZ+MZX+FiX, PZ+MYZ+FiY	6
Periodic + Radial	RiX+PX, RiY+PY, RiZ+ PZ	3
Periodic + Mirror + Radial	PX+MXY+RiX,PX+MZX+RiX, PY+MXY+RiY,PY+MYZ+RiY, PZ+MYZ+RiZ,PZ+MZX+RiZ	6
Periodic +Cyclic + Radial	PX+CX+RiX PY+CY+RiY, PZ+CZ+RiZ	3

11. In the above table:

FiX is one of FBX, FTX, FSX, FGX, F0X, SBX, STX or S2X

FiY is one of FBY, FTY, FSY, FGY, F0Y, SBY, STY or S2Y

FiZ is one of FBZ, FTZ, FSZ, FGZ, F0Z, SBZ, STZ or S2Z

RiX is one of RBX, RTX, RGX, or KX

RiY is one of RBY, RTY, RGY, or KY

RiZ is one of RBZ, RTZ, RGZ, or KZ

PX is one of PX or P0X

PY is one of PY or P0Y

PZ is one of PZ or P0Z

5

Examples: Mixing fabrication constraints

Here are possible combinations:

(MXY, MYZ AND MZX)

(CX AND MYZ) or (CY AND MZX) or (CZ AND MXY)

(EX, MXY AND MZX) or (EY, MYZ AND MXY) or (EZ, MZX AND MYZ)

(EX AND CX) or (EY AND CY) or (EZ AND CZ)

(FiX, MXY AND MZX) or (FiY, MYZ AND MXY) or (FiZ, MZX AND MYZ)

(FiX AND CX) or (FiY AND CY) or (FiZ AND CZ)

5.2.14 Topology Extra Variables

Topology design variables are automatically created as needed based on the TPROP and TSYM1/TSYM2/TSYM3 entries. Since these variables are generated internally, and have no user-meaningful identification, they cannot be accessed by the synthetic response facility. However, it is occasionally necessary to include design variables in synthetic responses, for example, in order to apply the beta method to define a min-max objective. For this reason, *GENESIS* provides the **TVAR** entry, which specifies extra design variables to be added to the topology optimization problem. These extra variables can be accessed by the TRESP2/TRESP3 entries.

The basic format for the extra design variable data in *GENESIS* is:

1	2	3	4	5	6	7	8	9	10
TVAR	ID	LABEL	INIT	LB	UB				

5.2.15 General Nonlinear Response Generation (TRESP2)

The **TRESP2** input data can be used to construct user defined responses that can be used as the objective function or can be constrained. These responses can be nonlinear functions of topology extra variables, tabled constants, and structural responses from different load cases. These user defined responses can be used, for example, to formulate the beta method of solving a min-max optimization problem. Many other types of responses can be generated. Since the TRESP2 capability is so flexible and powerful its use is only limited by the imagination of the user. The format of the TRESP2 data is:

1	2	3	4	5	6	7	8	9	10
TRESP2	ID	LABEL	EQID	REGION					
+	"TVAR"	NDV1	NDV2	NDV3	NDV4	NDV5	NDV6	NDV7	NDV8
+	Blank	NDV9	NDV10	...					
+	"DTABLE"	NC1	NC2	...					
+	"TRESP1"	NR1	NR2	...					

The alternate formats are

1	2	3	4	5	6	7	8	9	10
TRESP2	ID	LABEL	EQID	Blank					
+	"TVAR"	NDV1	NDV2	NDV3	NDV4	NDV5	NDV6	NDV7	NDV8
+	Blank	NDV9	NDV10	...					
+	"DTABLE"	NC1	NC2	...					
+	"TRESP1L"	NR1	LIDR1	NR2	LIDR2	NR3	LIDR3	...	

or

1	2	3	4	5	6	7	8	9	10
TRESP2	ID	LABEL	EQID						
+	"TVAR"	NDV1	NDV2	NDV3	NDV4	NDV5	NDV6	NDV7	NDV8
+	Blank	NDV9	NDV10	...					
+	"DTABLE"	NC1	NC2	...					
+	"TRESP2"	NS1	NS2	...					

The keywords TVAR, DTABLE, TRESP1 / TRESP1L and TRESP2 are used to indicate that the following data is extra variable ID's, tabled constant names, TRESP1 ID's or TRESP2 / TRESP3 ID's, respectively. More than eight ID's can be input using continuation lines (as shown for the TVAR ID's above). Not all types of ID's need be referenced. However, if more than one keyword is used, they must appear in the order shown above.

The TRESP2 data references **DEQATN** data which contains the equation that is used to calculate the response. The extra variable values, tabled constant values and structural response values are used as the variables in the equation.

As an example, consider the formulating the beta method to minimize the maximum z-component dynamic displacement magnitude of grid 22 over all the loading frequencies. This method requires introducing an extra variable, conventionally named beta, and is formulated as follows:

$$\begin{aligned} \min \beta \\ \text{subject to } \beta - |T_{322}(\Omega)| > 0 \end{aligned} \quad (\text{Eq. 5-18})$$

Note that the constraint is a function of a structural response, the dynamic displacement of the grid, which has different values at different loading frequencies; and an extra variable value, the added variable beta. It is advisable to scale the dynamic displacement in the constraint equation by its maximum value in the initial design. This is set by a tabled constant value:

	1	2	3	4	5	6	7	8	9	10
TVAR	1	BETA	1.0	0.0	1.0					
DTABLE	DMAX	2.0E-4								
TRESP1	40	G22C3	MDISP				3	22		
TRESP2	10	Constr	100							
+	TVAR	1								
+	DTABLE	DMAX								
+	TRESP1	40								
DEQATN	100	F(BETA,DMAX,G22C3) = BETA - G22C3/DMAX								
TCONS	10	1	0.0							
TRESP2	20	Obj	200							
+	TVAR	1								
DEQATN	200	F(BETA) = BETA								
TOBJ	20									

DEQATN 200 defines an identity function so that the value of the extra variable, beta, can be used as the objective function.

If the TRESP1 keyword is used in the TRESP2 data then the structural responses are all taken from the LOADCASE specified on the TCONS or TOBJ data for this TRESP2. Since all the responses are from the same LOADCASE, static, frequency and dynamic responses cannot be mixed.

Use of the TRESP1L keyword in the TRESP2 data allows responses from different LOADCASEs to be mixed. When the TRESP1L keyword is used a specific LOADCASE is listed after each TRESP1 ID. Since the LOADCASEs are specified in TRESP2 data, the LID field on the TCONS data must be left blank. The LOADCASE

specified on the TOBJ data must be the same as the first LOADCASE ID in the TRESP1L data (LIDR1). Also note that the REGION field on the TRESP2 must be left blank when TRESP1L data is used. As an example, consider a response that is the sum of the X direction displacement of GRID 10 in LOADCASE 3 and the first frequency in LOADCASE 6. The input data for this response would be:

	1	2	3	4	5	6	7	8	9	10
TRESP1	90	DISP1	DISP				1	10		
TRESP1	95	MODE1	FREQ				1			
TRESP2	200	DISPFRQ	990							
+	TRESP1L	90	3	95	6					
DEQATN	990	F(A,B)=A+B								
TCONS	200	34	55							

More details of the equation utility are given in the [Equation Utility](#) (p. 271).

5.2.16 Topology Variable Selection

In certain classes of problems it is desirable to have a fraction of the number of topology variables move to their upper bound while the rest to move to their lower bound. The **TSELECT** entry allows easy formulation of this type of “design variable selection” problem. The TSELECT entry can explicitly list a pool of topology extra variables or it can list a set of designable properties id to use the internally created topology design variables. The optimization process itself will find the optimal variables to move to their upper bound and the optimal variables to move to their lower bound. The optimization will be based on a user-given fraction along with an objective and, optionally, constraints. The objective and constraints could be from any *GENESIS* responses.

The format of DSELECT with topology extra variables is

1	2	3	4	5	6	7	8	9	10
TSELECT	ID	LABEL	FRACT	BTYPE	GTYPE	TOL			
+	“TVAR”	NDV1	NDV2	NDV3	NDV4	NDV5	NDV6	NDV7	NDV8
+		NDV9	-etc.-						

The ID specifies a unique TSELECT identification number. The LABEL field is to provide a brief description or name. The field for FRACT is to specify the desired fraction. FRACT has to be between 0.0 and 1.0. BTYPE must be either UPPER or EXACT or blank (Default is EXACT). GTYPE is to specify the type of constraint will *GENESIS* generate to enforce the fraction. GTYPE could be AVG or ABS. TOL is a small number that represents a tolerance used by the optimizer.

Example 1: Create constraints so that 50% of the listed topology extra variables move to their upper bound and 50% move to the lower bound.:

1	2	3	4	5	6	7	8	9	10
TSELECT	1		0.5	EXACT	AVG				
+	TVAR	87	25	3	8				

The format of TSELECT with internal topology variables is

1	2	3	4	5	6	7	8	9	10
TSELECT	ID	LABEL	FRACT	BTYPE	GTYPE	TOL			
+	“PROP”	PID1	PID2	PID3	PID4	PID5	PID6	PID7	PID8
+		PID9	-etc.-						

In this case PROP is used to indicate that the quantities that follow are property ids. The properties listed must be topology-designed. If a listed pid does not have associated **TPROP** data, *GENESIS* will issue a fatal error message.

Example 21: Create constraints so that 50% of the internally created topology design variables associated to properties 100, 200 and 300 move to their upper bound and 50% move to the lower bound.:

	1	2	3	4	5	6	7	8	9	10
TSELECT	1			0.5	EXACT	AVG				
+	PROP	100		200	300					

5.2.17 Progressive Rule

To help polarize topology results, the power rule value defined with RV1 in the **TPROP** data entry can be internally modified using the progressive rule option. The progressive rule will provide the value of RV1 used during the optimization process.

Turning Progressive Rule On/Off

The progressive rule can be turned on by setting to 1 the **DOPT** parameter **TCYCLEM**.

	1	2	3	4	5	6	7	8	9	10
DOPT										
+	TCYCLEM	1								

Valid values for TCYCLEM are 0 (off), 1(on) , -1, -2, -3, -4, -5 and -6.

The negative values correspond to built-in schedules. Schedules associated to -1 and -2 are mildly aggressive. Schedules associated to -3 and -4 are aggressive , while schedules associated to -5 and -6 are the most aggressive. Schedules -1, -3 and -5 use fewer design cycles than schedules -2, -4 and -6.

When the progressive rule is on, the program will use a progressive schedule defined internally. However, the internal schedule can be changed using the TCYCLE entry. By default, the progressive rule is turned off.

User-supplied Progressive Rule Schedule

The default progressive rule schedule (-2) can be changed by using the **TCYCLE** entry.

The basic format for TCYCLE is:

	1	2	3	4	5	6	7	8	9	10
TCYCLE										
+	DESMIN1	DESMAX2	RV1							
+	DESMIN1	DESMAX2	RV2							
+	DESMIN1	DESMAX2	RV3							
+	-etc.-									

Each continuation line in the TCYCLE entry defines a stage with the value of the topology power. The optimization process will use that constant value of RVi for at least DESMINi design cycles, but not more than DESMAXi design cycles. A stage may use fewer than DESMAXi cycles if the program meets the hard or soft convergence criteria, at which point it will move to the next stage. By default, the optimization process will not stop until all stages have been progressed through.

Example:

	1	2	3	4	5	6	7	8	9	10
TCYCLE										
+	2	8	1.80							
+	2	8	2.40							
+	2	8	3.00							

In the example above, the program will be used between 2 and 8 design cycles in each stage. In other words, the minimum number of design cycles the program will use with this schedule is 6 ($2+2+2$) while the maximum will be 24 ($8+8+8$).

Controlling Maximum Number of Design Cycles

When the progressive rules are used, the maximum number of design cycles that the software will need depends on the schedule used in TCYCLE. By default, *GENESIS* will use as many design cycles as are needed to complete all stages in the schedule. However, if the DOPT DESMAX field is used, *GENESIS* will stop when the number of design cycles completed reaches DESMAX.

If the DOPT parameter **DESMAXM** is set to 1, then the program will replace DOPT's DESMAX with the sum of all of the DESMAXi in the progressive schedule, thus ensuring that the schedule will be fully completed.

	1	2	3	4	5	6	7	8	9	10
DOPT	DESMAX									
+	DESMAXM	1								

If the DOPT parameter **DESMAXM** is set to 0, then the program will stop after reaching the DOPT's DESMAX design cycles, even if all the progressive stages have not been completed.

The default for DESMAXM is 1 when the DESMAX field is blank (or there is no DOPT entry). Otherwise, the default is 0.

Example 1: Do not exceed 5 design cycles:

	1	2	3	4	5	6	7	8	9	10
DOPT	5									
+	DESMAXM	0								

Example 2: Do not exceed 5 design cycles (Note: DESMAXM is not used here):

	1	2	3	4	5	6	7	8	9	10
DOPT	5									

Topology Optimization

Example 3: Use as many design cycle TCYCLE requires:

	1	2	3	4	5	6	7	8	9	10
DOPT										
+	DESMAXM	1								

Example 4 Use as many design cycle TCYCLE requires (Note|: field 2 of first line of DOPT is blank): :

	1	2	3	4	5	6	7	8	9	10
DOPT										

Reducing Number of Design Cycle by Controlling Grey Convergence

When the progressive rules is used, the number of design cycles used can be large. To reduce the number of design cycles, two DOPT parameters can be used: **CONVTS** and **CONVTD**.

Grey fraction is defined as the number of elements with intermediate density values divided by the total number of topology designable elements.

The CONVTS parameter is defined as the maximum change in the grey fraction for grey convergence between stages. If the all the constraints are satisfied and the change in the grey fraction value between the last design cycles of two consecutive stages is less than CONVTS, then the program is stopped without progressing through the remaining stages.

The CONVTD parameter is defined as the maximum change in the grey fraction for grey convergence between design cycles in last segment. If the all the constraints are satisfied and the change in the grey fraction value between two consecutive design cycles of the final stage in the schedule is less than CONVTD, then the program is stopped.

These parameters are only used with the progressive rule.

Example using DOPT parameters CONVTS and CONVTD

	1	2	3	4	5	6	7	8	9	10
DOPT	DESMAX									
+	CONVTS	0.001								
+	CONVTD	0.001								

The default for both CONVTD and CONVTS is 0.0. With the default value, the grey convergence check is not performed.

5.2.18 Hybrid Based Topology Method

In previous versions, there were two alternative topology methods to allocate the internally created topology design variables for a region. The first method is called element based topology method in while the second method is called geometry based topology method. The first method uses variables that are associated to the elements in the model while the second method uses variables that are mesh independent. The choice between which method to use is made automatically based on whether are design region specified a minimum member size or not. When minimum member size is used, the geometry method was selected.

As of GENESIS 17.0, an alternative to the geometry method is available: the hybrid method. Like the element method, the hybrid method internally creates design variables based on the element center locations but, like the geometry method, these variables also influence other elements based on distance.

Turning the Hybrid Method On/Off

When a topology region has no minimum member size specified, the element method is always used. When a minimum member size is given, the choice between the geometry method and the hybrid method is made by the DOPT parameter POLEM. If POLEM is 1, the hybrid method is selected, otherwise the geometry method is selected.

1	2	3	4	5	6	7	8	9	10
DOPT									
+	POLEM	1							

The default value for POLEM is 0.

Applicability

The hybrid method can only be used when minimum member size is specified on a TSYM1, TSYM2 or TSYM3 entry that has been referenced by TPROP. This method can work with mirror, cyclic, extrusion and periodic symmetries. Also cloning and maximum member size can be used with this method.

Currently, this method is not used on topology regions that include other types of fabrication constraints such as uniform, filling, stamping and radial.

5.2.19 Fully Polarized Results

Occasionally, it may be desirable to evaluate the final topology design using fully polarized results. This can be used, for example, to validate a design.

The **DOPT** parameter **TOPDIS** can be used to activate the polarized validation cycle. If this parameter is set to a value greater than 0.0 (but less or equal to 1.0), then *GENESIS* will perform one additional design cycle where all topology design variables are fully polarized.

The TOPDIS value is used to decide the cutoff design variable value. The **TOPDISM** parameter controls whether TOPDIS represent density fraction or Young's modulus fraction. A value of 0 (the default) will cause *GENESIS* to use TOPDIS as a mass fraction, while a value of 1 will use TOPDIS as a stiffness fraction. When TOPDISM is 0, then *TOPCUT* is defined to equal TOPDIS. If TOPDISM is 1 then *TOPCUT* is defined to equal $\text{TOPDIS}^{1/\text{RV1}}$, where RV1 is the (final) power of the power rule.

For the final polarized validation cycle, all topology design variables which are greater than or equal to *TOPCUT* will be set to 1.0, while the remaining topology design variables will be set to their lower bound (typically 0.0).

5.2.20 Use of Optimization Parameters

The **DOPT** data statement is used for selecting or changing default optimization parameters for shape/sizing optimization and for topology optimization. Some of these parameters are used for shape/sizing optimization only and some are used for topology optimization only. Some can be use for both types of optimization. The following table indicates for each parameter when it can be used.

DOPT PARAMETERS

Parameter Name	Shape/Sizing	Topology	Parameter Name	Shape/Sizing	Topology
ADJOIN	YES	NO	DSTART	YES	NO
BASIS	YES	NO	DVINIT2	YES	NO
ICOMPAPP	YES	NO	NDSCRT	YES	NO
ISHLAPP	YES	NO	IPEN	YES	NO
LAMASNS	YES	NO	PENLTD	YES	NO
IMATCH	YES	NO	PMULTD	YES	NO
DVINIT	YES	YES			
RMATCH	YES	NO	DDELA	YES	NO
OPTGRID	YES	NO	DDELC	YES	NO
FDCHU	YES	NO	DDELL	YES	NO
FDCHMU	YES	NO	DDELP	YES	NO
			DDAMIN	YES	NO
DELP	YES	NO	DDCMIN	YES	NO
DPMIN	YES	NO	DDLMIN	YES	NO
DELX	YES	YES	DDPMIN	YES	NO
DXMIN	YES	YES	ISDMET	YES	NO
DXFRAC	YES	NO	ISDMAX	YES	NO
			ISRMET	YES	NO
CONV1	YES	YES	ISRMAX	YES	NO
CONV2	YES	YES	OPTM	YES	YES
GMAX	YES	YES	IPRINT	YES	YES
CONVCN	YES	YES	BDMEM	YES	YES
CONVDV	YES	YES	ITRMOP	YES	YES
CONVLC	YES	NO	TINDEXM	NO	YES
CONVPR	YES	NO	FILTER	NO	YES
CONVPR	YES	NO	FILTNRM	NO	YES
CNTDC	YES	NO	TMIN	NO	YES
CNTIT	YES	NO	DELT	NO	YES
			DTMIN	NO	YES
DVTOL	YES	NO	DABOBJ	NO	YES
DVGTO	YES	NO	DELOBJ	NO	YES
PTOL	YES	NO	RPERT1	YES	NO
SGENEL	YES	NO	RPERT2	YES	NO
SK2UU	YES	NO	RCOMPAPP	YES	NO
KM2UU	YES	NO	RSHLAPP	YES	NO
			STRDOT	YES	NO
DINDEXM	YES	NO	DSCDOT	YES	NO
DR1VM	YES	NO	DNSHIS	NO	YES
MODAPP	YES	NO	OPTHIS	YES	NO
SYMMET	YES	NO	DVARHIS	YES	YES
SYMTOL	YES	NO	FREQREG	YES	YES

5.3 Topology Post Processing Result Files

Density Result File

The density result file is written into the *pname*DENSxx.ext file. For more information on the format of this file, see [Topology Density File \(pnameDENSxx.ext\)](#) (p. 843).

The format of this file corresponds to that selected with the executive control command POST. This file can be directly used in many postprocessing program such as Design Studio, Femb, Hypermesh, Patran or Ideas.

This file is used to visualize the topology optimization result at a selected design cycle xx.

Density History File

The density history file is written into the *pname*.DNS file. For more information on the format of this file, see [Topology Density File \(pname.DNS\)](#) (p. 842).

This file can be directly used in Femb. Use the history format to animate the design variable changes. These changes can be seen as density changes.

Isodensity File

The isodensity file is named *pname*TSURFxx.dat. This file uses *GENESIS* input data format so any preprocessor that can read *GENESIS* input files can read this post processing file. This output can only be generated for topology models designing shell and/or solid properties.

The isodensity levels can be controlled with the solution control command **TSURF**. A surface is generated such that it passes throughout the model wherever the element density percentage has the value specified by level, and encloses all regions where the density is greater than level. A smoothing algorithm is applied to this surface to produce the final output. The surface is defined by *GENESIS* GRID, CTRIA3, and CQUAD4 bulk data entries. Surfaces of different levels are output into the same file by using different property IDs for each level.

An isodensity file can be created during a normal run of *GENESIS* or after it. To create this file after a successful run, the executive control command **TSURFACE** is used. To use TSURFACE, the “*.HIS” file and in some occasions the “*.RST” file have to be available.

This file is used to visualize the last optimization cycle (or a given one, when using TSURFACE). This file is particularly useful for visualizing results of solid models.

5.4 Using the AUTORIB capability with Topology Optimization

GENESIS has a capability to automatically generate candidate rib stiffening elements for a given finite element model through the AUTORIB solution control command. AUTORIB is intended to be used in conjunction with topology optimization to determine the best places to add rib or bead stiffeners to a shell model.

The AUTORIB process follows these five steps:

1. Prepare the original model.
2. Run *GENESIS* in CHECK mode to generate the rib candidate elements.
3. Include the rib candidate elements back into the original data file and add topology data to design them.
4. Run *GENESIS* in TOPOLOGY mode to design the rib candidates.

Prepare the Original Model

A model is prepared for AUTORIB when it satisfies the following:

1. All surfaces that are to be studied are separated from the remainder of the model by referencing unique properties.
2. All CQUAD4 and CTRIA3 elements are oriented consistently.
3. A PROPSET bulk data entry (or entries) has been added to identify the surfaces AUTORIB should work with.
4. An AUTORIB solution control entry (or entries) has been added to specify the height and thickness of the rib candidates, and the isotropic material properties the rib elements should use.
5. The CHECK executive control statement has been added.

Run *GENESIS* in CHECK Mode

After running *GENESIS* in CHECK mode, using the prepared AUTORIB input data, the rib candidate element data is in the new *pname.RIB* output file.

Include Rib Candidate Data in Original Model

The rib candidate data is including back into the prepared model data file, either by copying and pasting with a text editor, or by simply adding "INCLUDE 'pname.RIB'" in the bulk data.

Many preprocessors do not correctly handle the INCLUDE command. To import the data into such preprocessors to view the rib candidates, the INCLUDE method cannot be used.

The AUTORIB solution control command should be removed or commented out.

Topology data to design the rib candidate elements should be added to the bulk data.

Important: The DOPT parameter **FILTER** should be set to the non-default value of 0 (zero) to turn off checkerboard filtering or -2 to reduce the influence of elements that are perpendicular to each other. If checkerboard filtering is not turned off, then the results will tend to make patches rather than directed ribs.

A relatively low mass fraction upper bound constraint should be used. Good results have been observed with mass fractions ranging from 0.05 to 0.10 (5% to 10%).

Run *GENESIS* in TOPOLOGY Mode

Running *GENESIS* in topology mode will result in the creation of the *pnameDENSxx.ext* or *pname.DNS* file, which contains the element densities.

5.5 Practical Recommendations

In this section practical recommendations are given for topology optimization

5.5.1 Do Not Mix Load Combinations with Topology Optimization

Since topology data cannot reference LOADCOMs the first practical rule is not to use LOADCOM with topology optimization. The LOAD bulk data entry can be used to produce linear combinations of load sets.

5.6 Example

Find the stiffest structure possible under one loading using 30% of the available material.

The topology data for this problem could be:

	1	2	3	4	5	6	7	8	9	10
\$										
\$ Topology designable regions										
\$										
TPROP	1	PSHELL	1	0.3	0.001					
+	POWER	3.0								
\$										
\$ Mass constraints										
\$										
TRESP1	10	MASS	MASSF R							
TCONS	10	ALL		0.30						
\$										
\$ Objective function										
\$										
TRESP1	100	OBJ	SENERGY							
TOBJ	100	ENERG Y								
\$										

In this example the design variables will start with a value of 0.3 to avoid starting infeasible. The minimum value of the design variables is 0.001. The TRESP1 10 is used to tag the system mass fraction and the TCONS is used to keep the mass under 30%.

The TRESP1 100 is used to tag the system strain energy and the TOBJ data is used to select the response 100 as the objective function of the problem.

In other words the problem is:

Minimize Strain Energy

Subject to;

$$\text{MASS} \leq 0.3\text{MASS}_{\text{ref}} \quad (\text{Eq. 5-19})$$

$$0.001 \leq X_i \leq 1.0 \quad (\text{Eq. 5-20})$$

$$E_i = 10^{-6}E_0 + (1 - 10^{-6})E_0X_i^{3.0} \quad (\text{Eq. 5-21})$$

$$\rho_i = \rho_0X_i \quad (\text{Eq. 5-22})$$

where,

MASS - Total mass

MASS_{ref} - Reference mass (mass when all design variables are 1.0)

E_i - Young's modulus associated to design variable i

E₀ - Initial Young's modulus (the value in MAT1)

ρ_i - Density of design variable i

ρ₀ - Initial density (the value in MAT1)

X_i - Topology design variable i

5.7 Mixing Topology with Shape, Sizing, Topometry, Topography and/or Freeform Optimization

Mixed Design Definition

Optimization problems may have both topology designable regions and parametric shape/sizing/topography/topometry/freeform designed entities at the same time. In other words, **TPROP** and/or **TVAR** topology data can be used together with DVAR, DVPROP1, DVPROP2, DVPROP3, DVPROP4, DVGRID, DVGRIDC, DOMAIN, DTGRID, DSPLIT and/or DSHAPE parametric data.

Mixed Responses

All response types from DRESP1 and **TRESP1** may be used in any optimization problem setup (i.e., topology-only, parametric-only or mixed). The optimization problem definition will include all constraints. If both topology and parametric objectives are defined, they will be combined into an index objective. In other words, DRESP1, DRESP2, DRESP3, DOBJ, DINDEX, DMATCH, DMATCH2, DCONS, DCONS2 and/or DSELECT can be used together with topology data **TRESP1**, **TRESP2**, **TRESP3**, **TOBJ**, **TINDEX**, **TCONS**, **TCONS2** and/or **TSELECT**.

Example:

A topology optimization problem that uses parametric entries:

1	2	3	4	5	6	7	8	9	10
\$									
\$ Topology designable regions									
\$									
TPROP	1	PSHELL	1	0.3					
\$									
\$ Objective function									
\$									
DRESP1	100	OBJ	SENERGY						
DOBJ	100	SENERGY							
\$ Massfr constraints									
\$									
TRESP1	20	MASSFR	MASSFR						
TCONS	20			0.30					

5.7.1 Mixing Topology and Sizing Optimization

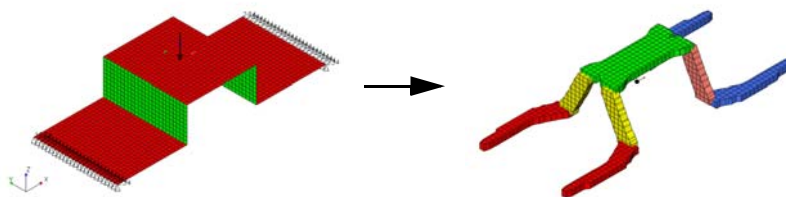
It is possible to mix topology and sizing optimization data. In fact, even the same property that is designed in sizing can be designed simultaneously with topology optimization.

Example:

A mixed topology and sizing optimization problem :

	1	2	3	4	5	6	7	8	9	10
\$										
\$ Topology designable region										
TPROP	1	PSHELL	1	0.3						
\$ Sizing designable region										
DVPROP3	1	1	SOLID							
+	10									
DVAR	10	THICK	1.0	0.1	3.0					
\$ Objective function										
DRESP1	100	OBJ	SENERGY							
DOBJ	100	SENERGY								
\$ Massfr constraints										
TRESP1	20	MASSFR	MASSFR							
TCONS	20			0.30						

The following picture shows the results of mixed sizing and topology optimization problem:



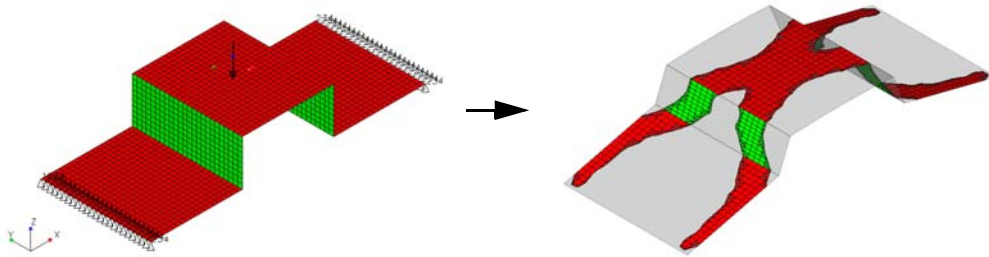
5.7.2 Mixing Topology and Topometry Optimization

It is possible to mix topology and topometry optimization data. However, the same property that is designed in topometry cannot be designed with topology optimization.

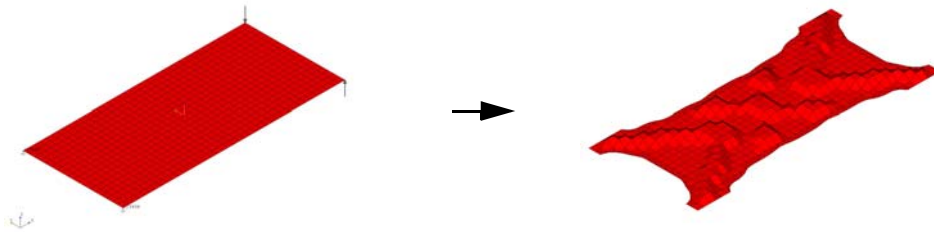
5.7.3 Mixing Topology and Shape, Topography and/or Freeform Optimization

It is possible to mix topology with shape, topography and/or freeform optimization data. In fact, the same elements that are topology designed can also be designed with shape, topography and/or freeshape.

The following figure shows the results of a mixed topology and shape optimization problem:



The following figure shows the results of mixed topology and freeform optimization problem:



5.7.4 Mixing Topology and Parametric Optimization: Mass versus Mass Fraction

In a typical topology-only optimization problem, mass fraction constraints are used to limit the amount of material used. On the other hand, in a typical parametric-only optimization problems, mass constraints are used. So questions that can arise are: What is better for a mixed problem: use MASS or MASSFR constraints?, What are the consequences of using MASS or MASSFR constraints? and What are the consequences of using both MASS and MASSFR constraints?

To help answer these questions: first a definition of mass fraction will be given and then some case studies are presented.

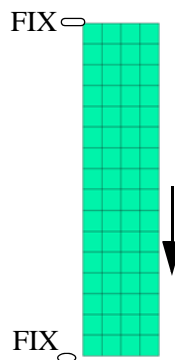
Mass Fraction Definition

Mass fraction is defined as the current mass divided by the mass the designable region would have if all of its topology design variables were set to 1.0. The mass the designable region would have if all of its topology design variables were 1.0 is equal to the nominal density, ρ_0 , times the volume of the designable region. If the volume of a topology designable region is increased or decreased by shape/sizing/topography/freeform, then it will take more or less total mass to have the same mass fraction response value.

Note that if there are no topology designed regions, then the mass fraction is always 1.0. As a consequence of this, mass fraction is not useful for non-topology problems.

Case Study: Comparing Mass and Mass Fraction Constraints

To understand the effect of using MASS, MASSFR or simultaneously both MASS and MASSFR, a simple problem is presented next. The problem is a simple rectangular plate which has its left corners fixed and it is subjected to a shear load its is right side midpoint.



Six cases are studied: 1a, 1b, 2a, 2b, 3a and 3b. In all six cases the objective function is to minimize the strain energy of the single loadcase. The constraints used in each case are described in the table below:

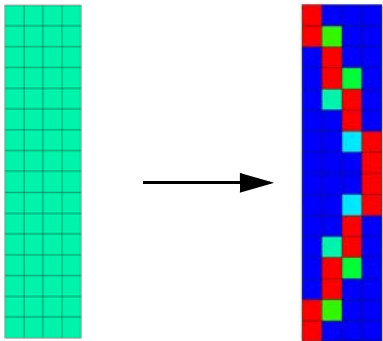
Case Studies		
Case Number	Optimization Type	Constraint Type
1a	Topology	MASS
1b	Mixed Topology and Shape	MASS
2a	Topology	MASSFR
2b	Mixed Topology and Shape	MASSFR
3a	Topology	MASS and MASSFR
3b	Mixed Topology and Shape	MASS and MASSFR

Case 1: Mass Constraint Only

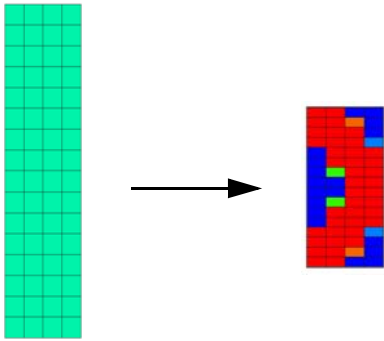
Assuming that a design starts with a density equal to 0.3 and that it is required to keep the final mass no larger than the initial mass, the following data can be used:

TPROP	1	PSHELL	1	0.3					
DRESP1	10	MASS	MASS						
DCONS2	10			1.0					

Case 1a: Topology Optimization



Case 1b: Topology and Shape Optimization



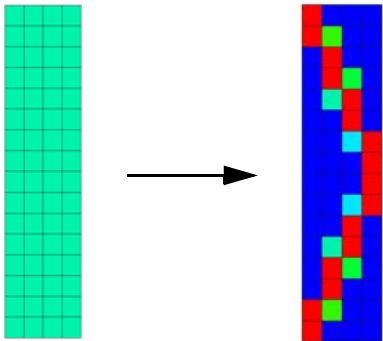
Shape optimization reduced the structure, but because in both optimization problems the same mass constraints were used, the final mass of both problems is the same. As a result of this, the mass fraction constraint associated to the second case is clearly higher. In general we can conclude that if parametric optimization reduces the size of the structure of a mixed problem, then the mass fraction of the resultant structure will be higher than its initial mass fraction, for the same mass. On the other hand, if parametric optimization increases the size of the structure, the mass fraction of the final structure will be lower than its initial mass fraction, for the same mass.

Case 2: Mass Fraction Constraint Only

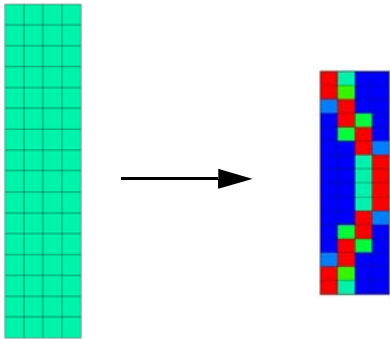
Assuming that a design starts with a density equal to 0.3 and that it is required to keep the mass fraction constant, the following data can be used:

TPROP	1	PSHELL	1	0.3					
TRESP1	20	MASSFR	MASSFR						
TCONS	20			0.3					

Case 2a: Topology Optimization



Case 2b: Topology and Shape Optimization



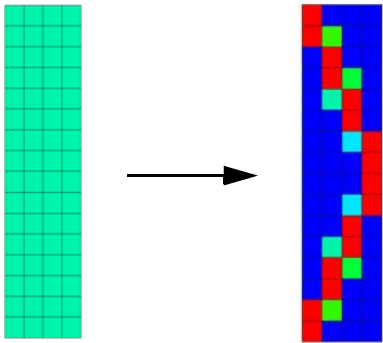
Shape optimization reduced the dimensions of the structure in 2b, but because both optimization problems used the same mass fraction constraint, the final mass of 2b problems is lower than the mass of 2a. In general we can conclude that if parametric optimization reduces the size of the structure, then the mass of the resultant structure will be lower than its initial mass, for the same mass fraction. On the other hand, if parametric optimization increases the size of the structure, the mass of the final structure will be higher than its initial mass, for the same mass fraction.

Case 3: Mass and Mass Fraction Constraints

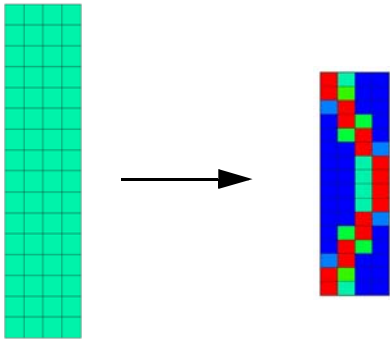
To simultaneously use mass and mass fraction constraints, the following data can be used:

TPROP	1	PSHELL	1	0.3					
DRESP1	10	MASS	MASS						
DCONS2	10			1.0					
TRESP1	20	MASSFR	MASSFR						
TCONS	20			0.3					

Case 3a: Topology Optimization



Case 3b: Topology and Shape Optimization



Both optimization problems used the same mass fraction constraint and the same mass constraint, but because shape optimization reduced the structure in 3b the final answer was dominated by the mass fraction constraint.

In general, it can be conclude that, in mixed topology and parametric optimization problems when both mass and mass fraction constraints are used, if the parametric optimization reduces the size of the structure, then the mass fraction constraints will dominate and the mass of the resultant structure will be lower than its initial mass fraction. On the other hand, if parametric optimization increases the size of the structure the mass constraint will dominate and the mass of the final structure will be same as the initial mass.

CHAPTER 6

Executive Control

- ANALYSIS
- DLIB
- DRESP3
- GUSER
- HIS
- RESTART
- RST
- SCRIPT
- SENSITIVITY
- SSOLID
- TOPOLOGY
- TSURFACE

6.1 ANALYSIS

Executive Control Entry: **ANALYSIS** - Program Flow Control

Description: Stops the program after the first analysis.

Format:

ANALYSIS

Example:

ANALYSIS

Remarks:

1. The following executive control commands are mutually exclusive (only one may be specified): ANALYSIS, CHECK, REDUCE, **SENSITIVITY**, **SSOLID**, **TSURFACE**
2. *GENESIS* will perform the analysis after updating the model. In other words, the design data will be used to update grid and property data if there is shape/sizing data, and/or update material data if there is topology data.
3. ANALYSIS can also be specified on the command line using the -analysis flag. For example.

```
genesis -analysis mydata.dat
```

If any command mutually exclusive with ANALYSIS appears in the datafile, it takes precedence over any value specified on the command line.

6.2 DLIB

Executive Control Entry: **DLIB** - External User Library Control

Description: Defines the name of an external shared object (DLL) that will calculate user-defined properties and stresses for the DVPROP3 bulk data entry.

Format:

DLIB = *lid*, full_path_to_shared_object

Alternate Format:

DLIB = full_path_to_shared_object

Examples:

DLIB = 20, /home/mrt/work/dlib.so

DLIB = 16, D:\users\mrt\dlib.dll

Option	Meaning
--------	---------

<i>lid</i>	Unique DLIB library identification number (Integer >14).
------------	--

Remarks:

1. If no library ID is specified, the library will be used to handle all IDs not specifically defined. At most one DLIB command without a library ID is allowed.
2. This command may not be available on all operating systems.
3. One or more **DLIB** bulk data entries must exist to make the user cross-sections available for use with **DVPROP3**.
4. The shared object must export two required interface functions. The interface functions have the following Fortran declarations:

```
SUBROUTINE DLIBPROP(IWHICH, CSD, V, SP, IERR)
  INTEGER IWHICH, IERR
  DOUBLE PRECISION CSD(*), V, SP(*)
```

```
SUBROUTINE DLIBSTRESS(IWHICH, CSD, FORCE, STRESS, IERR)
  INTEGER IWHICH, IERR
  DOUBLE PRECISION CSD(*), FORCE(*), STRESS(*)
```

Note that if a language other than Fortran is used to create the shared object, care must be taken to ensure that the correct interface function names are exported. The actual required function names are system dependent. For example, using the C language, the functions should be named as follows:

Microsoft Windows	DLIBPROP DLIBSTRESS
Solaris, Linux, HP-UX	dlibprop_ dlibstress_
AIX	dlibprop dlibstress

For more details see [Adding to the Design Element Library](#) (p. 285).

6.3 DRESP3

Executive Control Entry: **DRESP3** - External Response Control

Description: Defines the name of an external shared object (DLL) that will calculate external user-defined responses for the DRESP3 or TRESP3 bulk data entry.

Format:

DRESP3 = *lid*, full_path_to_shared_object

Alternate Format:

DRESP3 = full_path_to_shared_object

Examples:

DRESP3 = 2, /home/mrt/work/dresp3.so

DRESP3 = 17, D:\users\mrt\dresp3.dll

Option	Meaning
<i>lid</i>	Unique DRESP3 library identification number (Integer >0).

Remarks:

1. If no library ID is specified, the library will be used to handle all IDs not specifically defined. At most one DRESP3 command without a library ID is allowed.
2. This command may not be available on all operating systems.
3. **DRESP3** or **TRESP3** bulk data entries must exist to request external user-defined responses.
4. The shared object must export one required interface function. The interface function has the following Fortran declaration:

```
SUBROUTINE DRESP3(IWHICH, VAR, N, VAL, IERR)
  INTEGER IWHICH, N, IERR
  DOUBLE PRECISION VAR(*), VAL
```

Note that if a language other than Fortran is used to create the shared object, care must be taken to ensure that the correct interface function name is exported. The actual required function name is system dependent. For example, using the C language, the function should be named as follows:

Microsoft Windows	DRESP3
Solaris, Linux, HP-UX	dresp3_
AIX	dresp3

For more details see [Use of the DRESP3 / TRESP3 capability](#) (p. 307).

6.4 GNUSER

Executive Control Entry: **GNUSER** - External Response Control

Description: Defines the name of an external program that will calculate user-defined responses.

Format:

GNUSER = executable_program_name

Example:

GNUSER = ./myresp_prog

Remarks:

1. Only one GNUSER command is allowed.
2. **DRESPU** bulk data entries must exist to request external user-defined responses. For more details see **DRESPU** (p. 297).

6.5 HIS

Executive Control Entry: **HIS** - Restart Control

Description: Defines the name of an alternative history file to be used in a restart run.

Format:

```
HIS 'file name'
```

Alternate Format:

```
HIS = file name
```

Examples:

```
HIS '../run1/project.HIS'
```

```
HIS = C:\saved_files\projA001.HIS
```

Option	Meaning
--------	---------

file name	External file name. The user must provide the file name according to the machine installation.
-----------	--

Remarks:

1. This command is optional. At most one HIS command is allowed.
2. Restart runs will read design variable values from the *pname.HIS* file, where *pname* is the project name of the current run. If this command is present, then at the start of the run, the specified file will be copied to *pname.HIS*. This command will not cause new history data to be written to the specified file.
3. The file name is limited to 240 characters.
4. If the quoted format is used, and the line does not end with a quote character, additional lines will be read until the closing quote is found. Leading and trailing spaces on continued and continuation lines are discarded.

6.6 RESTART

Executive Control Entry: **RESTART** - Program Flow Control

Description: Uses results from a previous optimization run to reset the design variable initial values.

Format:

RESTART = n,m

Alternate Format:

RESTART = LAST,k

Examples:

RESTART = 7

RESTART = LAST, 5

Option	Meaning
n	The design cycle from a previous run to restart from (Integer>0).
m	The new maximum number of design cycles (Integer>n or blank). Default is the value specified on the DOPT bulk data entry.
LAST	Word indicating to restart from the last design cycle in the *.HIS file.
k	The maximum number of <u>additional</u> design cycles (Integer>0 or blank). Default is 10.

Remarks:

1. The value m is the maximum total number of design cycles, not the maximum number for this run. That is, m includes the design cycles from the previous run(s). Conversely, when the alternate format is used, the value k is the maximum design cycles for this run only.
2. See **Restart from any Previous Design Cycle** (p. 322) for a more in depth discussion of the restart capability available in *GENESIS*.
3. RESTART can also be specified on the command line using the -restart=n,m flag. For example.

`genesis -restart=10,20 mydata.dat`

If RESTART appears in the datafile, it takes precedence over any value specified on the command line.

6.7 RST

Executive Control Entry: **RST** - Restart Control

Description: Defines the name of an alternative restart file to be used in a restart run.

Format:

```
RST 'file name'
```

Alternate Format:

```
RST = file name
```

Examples:

```
RST '../run1/project.RST'
```

```
RST = C:\saved_files\projA001.RST
```

Option	Meaning
--------	---------

file name	External file name. The user must provide the file name according to the machine installation.
-----------	--

Remarks:

1. This command is optional. At most one RST command is allowed.
2. Restart runs that contain DCONS2, DINDEX, TCONS2 or TINDEX data will read response normalizing data from the *pname*.RST file, where *pname* is the project name of the current run. If this command is present, then at the start of the run, the specified file will be copied to *pname*.RST. This command will not cause new restart data to be written to the specified file.
3. The file name is limited to 240 characters.
4. If the quoted format is used, and the line does not end with a quote character, additional lines will be read until the closing quote is found. Leading and trailing spaces on continued and continuation lines are discarded.

6.8 SCRIPT

Executive Control Entry: **SCRIPT** - Customization Control

Description: Defines a customization script to alter normal program behavior.

Format:

```
SCRIPT = engine, timeout
script lines
.
.
ENDSCRIPT
```

Example:

```
SCRIPT = LUA
genesis.hook.READ = function()
    genesis.diag[1] = 2 -- same as DIAG=12
end
ENDSCRIPT
```

Option	Meaning
<i>engine</i>	Script engine. Word 'LUA'.
<i>timeout</i>	Script timeout (in secs). If any script function takes longer than timeout, the script engine will abort and no further script functions will run. (Integer > 0. Default = 900).
<i>script lines</i>	Script definition with syntax defined by the script engine.

Remarks:

1. At most one script definition may appear in an input file.
2. If the SCRIPT entry occurs in an auxilliary file added with INCLUDE, and the end of that file is reached while reading *script lines*, an implicit ENDSCRIPT will automatically be inserted to end the SCRIPT entry before continuing to read lines from the main input file.
3. If the script encounters an error condition during processing, the entire *GENESIS* run will abort.
4. See [Customization Through Scripts](#) (Sec. 4.16) for an in-depth discussion of the scripting capability available in *GENESIS*.
5. Scripts should be used with caution. Scripts change normal program behavior and may cause unexpected results or errors.

6.9 SENSITIVITY

Executive Control Entry: **SENSITIVITY** - Program Flow Control

Description: Stops the program after the first sensitivity calculation.

Format:

SENSITIVITY

Example:

SENSITIVITY

Remarks:

1. The following executive control commands are mutually exclusive (only one may be specified): **ANALYSIS**, **CHECK**, **REDUCE**, **SENSITIVITY**, **SSOLID**, **TSURFACE**
2. The solution control command, **SENSITIVITY**, must also be used if this keyword is used.
3. Sensitivity results are only available for shape/sizing optimization. This command is not intended to be used in a topology optimization run.
4. **SENSITIVITY** can also be specified on the command line using the -sensitivity flag. For example.

```
genesis -sensitivity mydata.dat
```

If any command mutually exclusive with **SENSITIVITY** appears in the datafile, it takes precedence over any value specified on the command line.

6.10 SSOLID

Executive Control Entry: **SSOLID** - Program Flow Control

Description: Uses results from a previous optimization run the reset the design variable initial values, and only performs SSOL output generation.

Format:

SSOLID = n

Example:

SSOLID = 7

Option	Meaning
--------	---------

n	The design cycle from a previous run to use (Integer>0).
---	--

Remarks:

1. The following executive control commands are mutually exclusive (only one may be specified): **ANALYSIS**, CHECK, REDUCE, **SENSITIVITY**, SSOLID, **TSURFACE**
2. The results of this run are stored in the *pname*SSOLxx.dat. Where, *pname* is the project name and *xx* corresponds to the design cycle number.
3. The SSOL file contains grid data and solid element connectivity (CHEXA and CPENTA) that reveal the thicknesses of CQUAD4 and/or CTRIA3 elements of the input data file. This command causes only SSOL output generation to be performed, with no analysis or optimization. The solution control command **SSOL** must also be used to request the SSOL output generation.
4. To use this command, *GENESIS* needs the *pname*.HIS file from a previous *GENESIS* run.

6.11 TOPOLOGY

Executive Control Entry: **TOPOLOGY** - Optimization Data Control

Description: Control mixing of shape/sizing parametric design data and topology design data.

Format:

$$\text{TOPOLOGY} = \begin{Bmatrix} \text{YES} \\ \text{NO} \end{Bmatrix}$$

Example:

TOPOLOGY = YES

Remarks:

1. This command is optional. If this command is not present, then all optimization data in the input file will be used.
2. If “TOPOLOGY = YES” is present, then only topology design data will be used. Parametric design bulk data (**DCONS**, **DCONS2**, **DINDEX**, **DLINK**, **DMATCH**, **DMATCH2**, **DOBJ**, **DOMAIN**, **DRESP1**, **DRESP2**, **DRESP3**, **DRESPG**, **DRESPU**, **DSELECT**, **DSHAPE**, **DSPLIT**, **DTGRID**, **DVAR**, **DVGRID**, **DVGRIDC**, **DVPROP1**, **DVPROP2**, **DVPROP3**, **DVPROP4**, **DVSET** and **DVSET1**) will be ignored.
3. If “TOPOLOGY = NO” is present, then only parametric design data will be used. Topology design bulk data (**TCONS**, **TCONS2**, **TINDEX**, **TOBJ**, **TPROP**, **TRESP1**, **TRESP2**, **TRESP3**, **TSELECT**, **TSYM1**, **TSYM2**, **TSYM3** and **TVAR**) will be ignored.
4. The TOPOLOGY command may appear at most once. TOPOLOGY = YES may not be combined with executive control command **SENSITIVITY**,

6.12 TSURFACE

Executive Control Entry: **TSURFACE** - Program Flow Control

Description: Uses results from a previous topology run the reset the design variable initial values, and only performs TSURF output generation.

Format:

TSURFACE = n

Example:

TSURFACE = 17

Option	Meaning
--------	---------

n	The design cycle from a previous run to use (Integer>0).
---	--

Remarks:

1. The following executive control commands are mutually exclusive (only one may be specified): **ANALYSIS**, CHECK, REDUCE, **SENSITIVITY**, **SSOLID**, **TSURFACE**
2. TSURFACE results are only available for topology optimization results.
3. The results of this run are stored in the *pname*TSURFxx.dat. Where, *pname* is the project name and xx corresponds to the design cycle number.
4. The isodensity file contains smoothed surface meshes. This command causes only density smoothing to be performed, with no analysis or optimization. The solution control command **TSURF** should be used to specify the desired levels for density smoothing.
5. To use this command *GENESIS* needs the *pname*.HIS file and possibly the *pname*.RST file from a previous *GENESIS* run.

CHAPTER 7

Solution Control

- Natural (Displacement) Basis (Perturbation) Vectors
- Grid Basis Vectors
- General Design Output Control Commands
- Shape, Sizing, Topography, Topometry and Freeform Output Control Commands
- Topology Output Control Commands
- Output Selection
- Solution Control Data

7.1 Natural (Displacement) Basis (Perturbation) Vectors

The displacements produced in a set of grids by loads in a static loadcase or loadcom can be used to automatically generate basis or perturbation vectors for shape optimization. The command **DVSHAPE** is used for this purpose. For example;

```
LOADCASE 10
  LABEL = NATURAL BASIS VECTOR 1
  LOAD = 100
  SPC = 20
  DVSHAPE = 10, 1.0, 2.0, ALL, 1.0
SET 100 = 1, 2, 3, 4, 5, 6
SET 200 = 7, 8, 9, 10, 11, 12
LOADCASE 20
  LABEL = NATURAL BASIS VECTOR 2
  LOAD 200
  DVSHAPE = 20, -1.0, 1.0, 100, 1.0
  DVSHAPE = 40, -1.0, 1.0, 200, 2.0
```

In this example, the DVSHAPE in the first LOADCASE will produce one design variable data and one DVGRID data per grid in the structure that has non zero displacements. The DVSHAPE data in LOADCASE 20 will produce 2 design variables and DVGRID data on grids belonging to SET 100 and SET 200 that have non zero displacements. For more details on DVSHAPE, see [Natural Perturbation Vectors \(DVSHAPE Data\)](#) (p. 182).

The DVAR and DVGRID data created using this command are stored in the *.DVS file.

7.2 Grid Basis Vectors

The GRID coordinates listed in the bulk data can be used to automatically generate basis vectors for shape optimization. The command **DVBASIS** is used for this purpose. For example;

```
DVBASIS = 10, 1.0, 2.0, ALL
SET 100 = 1, 2, 3, 4, 5, 6
SET 200 = 7, 8, 9, 10, 11, 12
DVBASIS = 20, -1.0, 1.0, 100
DVBASIS = 40, -1.0, 1.0, 200
```

In this example, the first DVBASIS will produce one design variable data and one DVGRID data per grid in the structure. The DVBASIS 20 and 40 will produce 2 design variables and DVGRID data on grids belonging to SET 100 and SET 200. For more details on DVBASIS, see [Grid Basis Vectors \(DVBASIS Data\)](#) (p. 187).

This command should be used in an auxiliary input file where the grids are perturbed from the original grids. The grids in the auxiliary input data file should represent candidate designs.

7.3 General Design Output Control Commands

The command **APRINT** is used to control the printing of analysis results during the design process. If APRINT=ALL, the default, then analysis results are printed for each design cycle. If APRINT=FIRST the analysis results are only printed for the initial design. If APRINT=LAST the analysis results are only printed for the final design. If APRINT=FLAST the analysis results are printed for the initial and final designs. If APRINT=n then the analysis results are printed for the initial design, every nth design after the initial design, and the last design. If APRINT=NONE then no analysis results are output.

The command **DPRINT** is similar to the command APRINT except that, instead of controlling printing of the analysis results, it controls printing of the design results.

The command **OPRINT** is similar to the command APRINT except that it controls the frequency of production of optimization result post-processing files (*pname*DENSxx.ext, *pname*OPOSTxx.ext and *pname*.SHP).

The command **MPRINT** is used to control the printing of the matrices associated with Guyan reduction loadcases. If MPRINT = ALL, the default, then the matrices are printed for each design cycle. If MPRINT = FIRST, the matrices are printed only for the first design cycle. If MPRINT = NONE, the matrices are not printed. Note that the matrices associated with Guyan Reduction are KAA: the reduced stiffness matrix, MAA: the reduced mass matrix, KAASENS: the sensitivity of KAA and MAASENS: the sensitivity of MAA.

The command SUMMARY controls the printing of a summary table of the analysis and design problem sizes. SUMMARY = YES, the default, is used to request printing of the table. SUMMARY = NO is used when no printing is desired.

The command **DRESP2** is used to control the printing of the synthetic response values at each design cycle. DRESP2 = YES is the default.

The command **SENSITIVITY** is used to control the printing of the sensitivities of the retained design responses at each design cycle. SENSITIVITY=PRINT causes the sensitivities to be printed to the output file. SENSITIVITY = POST causes the sensitivities to be written to the “.SEN” post processing file. SENSITIVITY = BOTH causes both types of output to be produced. SENSITIVITY = NONE, the default, causes no gradient output to be produced.

The command **AUTORIB** generates and prints rib-stiffener elements at candidate locations over the entire surface specified by a property set. These elements can be used in a subsequent topology optimization to determine the best place to add rib and/or bead stiffening.

7.4 Shape, Sizing, Topography, Topometry and Freeform Output Control Commands

The command **DESIGN** is used to control the printing of the updated properties and grid locations. If **DESIGN** = **PROP**, the updated properties are printed. If **DESIGN** = **GRID**, the updated grid locations are printed. If **DESIGN** = **BOTH**, the updated properties and grid locations are both printed. Finally, if **DESIGN** = **NONE**, the default, then no updated properties or grid locations are printed.

The command **DVGRID** is used to output all the automatic DVGRID data generated by *GENESIS*. This includes the data generated by DVGRIDC-DOMAIN data statements and DTGRID. The syntax is DVGRIDC=PRINT to output the DVGRID data and DVGRID=NONE to avoid printing. In addition to the DVGRID data, if applicable, the generated DVAR and DLINK data will be also output.

The command **UPRINT** is used to control the printing of the updated analysis input data file. If **UPRINT** = **ALL**, an updated file is printed for every design cycle. If **UPRINT** = **FLAST**, an updated file is printed for the first and last design cycles. If **UPRINT** = **FIRST**, **LAST** or **n**, then the updated file is printed for the first design cycle or the last design cycle or every **n** design cycles. Finally, if **UPRINT** = **NONE**, the default, then no updated properties or grid locations are printed. Only the analysis data is printed.

The command **SIZING** is used to control the printing of designed element properties during the optimization process. The command **SIZING** = **POST** will generate the *pnameOPOSTxx.ext* file, containing element property values, along with the relative and absolute changes in value, in the element strain energy analysis post processing format. The command **SIZING**=**NONE** can be used to avoid the printing.

The OPOST file can be used to visualize the results of sizing and/or topometry optimization by coloring elements according to the property value.

The commands **BASIS** and **PERTURBATION** are synonymous. Either of these can be used to output the basis or perturbation vectors in post processing format (**PUNCH**). The syntax is **BASIS** = **POST** or **PERTURBATION** = **POST** to get the post processing file (extension is DVG). The syntax is **BASIS** = **NONE** or **PERTURBATION** = **NONE** to avoid the printing. The default is **NONE** for both commands.

The *.DVG file can be used to animate each of the basis or perturbation vectors. The files are written using the eigenvector format, so almost any post-processing program can be used to see the animation.

The command **SHAPE** is used to control the printing of the shape changes during the optimization process. The command **SHAPE** = **POST** causes *GENESIS* to print the difference between the current shape and the original shape as displacements, using the **PUNCH** format (extension is SHP). The syntax is **SHAPE** = **NONE** to avoid printing.

Solution Control

The *.SHP file can be used to animate the shape optimization process. The file is written using the displacement format so almost any post-processing program that supports transient files can be used to see the animation of the shape optimization process. If mesh smoothing is used, the parameter PSMOOTH = YES will produce the *.SHP file to include the changes due to mesh smoothing.

The command **DVSHAPE** is used to control the generation and printing of the Natural (displacement) Basis (Perturbation) vectors. The command DVSHAPE = DVID, LB, UB, SETID, maxp causes *GENESIS* to print the associated vectors and DVARs using the *GENESIS* format for DVGRID and DVAR in the *pname.DVS* file.

The *.DVS file can be used in a new input data file to use the created data by either copying it into the new file or by using the INCLUDE '*pname.DVS*' command.

The command **DVBASIS** is used to control the generation and printing of the grid basis vectors. The command DVBASIS = DVID, LB, UB, SETID causes *GENESIS* to print the associated vectors and DVARs using the *GENESIS* format for DVGRID and DVAR in the *pname.DVB* file.

The *.DVB file can be used in a new input data file to use the created data by either copying it into the new file or by using the INCLUDE '*pname.DVB*' command.

The command **GRAPH** is used to control the printing of graphs containing the objective function and the maximum constraint violation versus design cycle. The extension of this file is **ps** or **html** depending on the selected format. The syntax is GRAPH = YES or NO. The default is YES. The format of this file is postscript or html.

7.5 Topology Output Control Commands

The command **DENSITY** is used to control the printing of element topological densities. The command `DENSITY = POST` will generate the *pnameDENSxx.ext* file, containing element normalized densities, in the element strain energy analysis post processing format. The command `DENSITY=NONE` can be used to avoid the printing.

The command **TSURF** is used to convert topology densities into a *GENESIS* input file that contains meshes that represent isodensity surfaces. The results of this command are written in the *pnameTSURFxx.dat* file. *pname* is the project name and *xx* corresponds to the design cycle number.

The *pnameTSURFxx.dat* file can be used to visualize the topology results. This file is written in *GENESIS* input data format so it can be used in any preprocessor that can read *GENESIS* files.

7.6 Output Selection

Output Control

APRINT	Controls the analysis results output printing
DPRINT	Controls the design results output printing.
MPRINT	Controls the printing of the reduced matrices used in Guyan reduction
OPRINT	Controls the optimization post-processing file production.
SUMMARY	Controls the printing of analysis and design summary tables
DRESP2	Controls the printing of the synthetic response values
SENSITIVITY	Controls the printing of sensitivity data

Guyan Reduction Matrices Output Requests

KAASENS	Requests the printing of the sensitivity of the Guyan reduced stiffness matrix
MAASENS	Requests the printing of the sensitivity of the Guyan reduced mass matrix

Natural Basis (Perturbation) Vector Output Requests

DVSHAPE	Requests the printing of a set of displacements as basis or perturbation vectors using the DVGRID format
----------------	--

GRID Basis Vector Output Requests

DVBASIS	Requests the printing of a set of grids as basis vectors using the DVGRID format
----------------	--

Design Variable Output Requests

TVAR	Controls the printing of the all design variables.
-------------	--

Shape, Sizing, Topography and Topometry Output Requests

DESIGN	Controls the design results output printing.
UPRINT	Controls the printing of updated input file.
DVGRID	Controls the printing of automatically generated DVGRID/DVAR and DLINK data.
GRAPH	Controls the printing of the graph that contains the objective and maximum constraint violation histories
PERTURBATION	Controls the printing of PERTURBATION vectors in a post-processing file.
SHAPE	Controls the printing of shape changes in a PUNCH or PATRAN post-processing file.
SIZING	Controls the printing of element property values in a post-processing file for visualization.
SSOL	Controls the printing of the shell elements as solid elements for the purpose of visualization the thickness of the shells
THICKNESS	Controls the printing of the shell element thickness in a postprocessing file for visualization.

Topology Output Requests

DENSITY	Controls the printing of the density in a postprocessing file.
DESIGN	Controls the printing of updated materials.
GRAPH	Controls the printing of the graph that contains the objective and maximum constraint violation histories
TSURF	Controls the printing of the isodensity surfaces

7.7 Solution Control Data

The format of the Solution Control data is free-field. In presenting general formats for each entry embodying all options, the following conventions are used:

1. Upper-case letters must be typed as shown.
2. Lower-case letters indicate that a substitution must be made.
3. Braces { } indicate that a choice of contents is mandatory.
4. Brackets [] contain an option that may be omitted or included by the user.
5. Bold options or values are the default values.
6. Physical data entry consists of information input in columns 1 through 72 of a data entry. Most Solution Control data is limited to a single physical entry.
7. Logical entry may have more than 72 columns with the use of continuation data.
8. Comment lines can be input by using the dollar sign (\$) in the first column of the line.

If the first four characters of a mnemonic are unique relative to all other Solution Control entries, the characters following can be omitted.

7.7.1 APRINT

Solution Control Entry: **APRINT** - Analysis Output Control

Description: Requests analysis results output only at selected design cycles

Format:

$$\text{APRINT} = \left\{ \begin{array}{c} \text{ALL} \\ \text{FIRST} \\ \text{LAST} \\ \text{FLAST} \\ n \\ \text{NONE} \end{array} \right\}$$

Examples:

APRINT = ALL

APRINT = 2

Option	Meaning
ALL	Default. Analysis results output at every Design Cycle.
FIRST	Analysis results output only at the first Design Cycle.
LAST	Analysis results output only at the last Design Cycle.
FLAST	Analysis results output only at the first and last Design Cycle.
n	Analysis results output for the initial, every n-th design cycle after the initial, and the final design cycle (Integer > 0).
NONE	No analysis results are printed.

Remarks:

1. To control printing of DRESP2/3 and TRESP2/3 response values, use the solution control command **DRESP2**.

7.7.2 AUTORIB

Solution Control Entry: **AUTORIB** - Automatic Rib Candidate Output Request

Description: Request generation and printing of rib-stiffener candidate elements for selected PSHELL/PCOMP property sets.

Format:

AUTORIB = propset,height,thick,e,nu,rho,nelht,qdiag

Examples:

AUTORIB = 10,6.0,0.5,2.07E5,0.3,7.8E-9

AUTORIB = 10,1.0,0.1,30.0E6,0.3,7.3E-4,0

Option	Meaning
propset	Set identification number of PROPSET entry or entries in the bulk data (Integer>0).
height	Height of rib. (Real > 0.0).
thick	Thickness of rib. (Real > 0.0).
e	Young's modulus for rib elements. (Real > 0.0)
nu	Poisson's ratio for rib elements. (Real > 0.0)
rho	Material density for rib elements. (Real > 0.0)
nelht	Number of elements through the height. (Integer ≥ 0 or blank) (Default = 1)
qdiag	Flag indicating whether to make candidates across quad element diagonals. (Integer 0 or 1 or blank) (Default = 1)

Remarks:

1. AUTORIB generates rib-stiffeners at candidate locations over the entire surface specified by propset. These elements can be used in a subsequent topology optimization to determine the best place to add rib and/or bead stiffening. Multiple AUTORIB entries are allowed. Each AUTORIB entry creates one new material and one new property for candidate locations specified by the property set.
2. Candidate locations are: 1) Edges between CQUAD4/CTRIA3 elements in the specified property set; 2) Diagonals across CQUAD4 elements in the specified property set (if the qdiag parameter is 1). If more than two elements touch an edge, then that edge is removed from the candidate location set. If the normal directions of two elements sharing an edge are too different, then that edge is removed from the candidate location set. For this reason, it is important that the normals of all elements in the property set be oriented consistently. CQUAD4 rib candidate elements are generated on the positive normal side of the elements in the property set.

3. Only PSHELL and/or PCOMP properties in the property set specified by propset will be used by AUTORIB. If the specified propset contains no PSHELL/PCOMP properties, no rib candidates will be generated.
4. The nelht parameter specifies the number of elements to generate through the height of the rib. If nelht is 0, then one CBAR element will be generated at each rib candidate location. If nelht > 0, then nelht CQUAD4 elements will be generated at each rib candidate location.
5. The height of a generated rib candidate may need to be reduced on curved surfaces to prevent malformed elements.
6. This command creates MAT1, GRID, and PSHELL/CQUAD4 or PBAR/CBAR data. The generated data is stored in the *pname*.RIB file. The rib candidate data is used by adding the bulk data command: INCLUDE '*pname*.RIB' to the original data file. Alternatively, the user may cut and paste the results in the RIB file to its new input file.
7. The data generated by AUTORIB is only printed to the *pname*.RIB file, and is not used for any analysis results. It is recommended that AUTORIB be used only in conjunction with the CHECK executive control command.
8. The AUTORIB module attempts to use identification numbers for the generated data that are unique with respect to the existing data. However, if the existing data uses high numbers (8 digits) for ids, then the generated ids may not be unique.

7.7.3 BASIS

Solution Control Entry: **BASIS** - Shape and Sizing Design Output Request

Description: Requests writing the basis vector for each shape design variable in a post processing file.

Format:

$$\text{BASIS} = \begin{Bmatrix} \text{POST} \\ \text{NONE} \end{Bmatrix}$$

Example:

`BASIS = POST`

Option	Meaning
POST	Basis vectors are printed in the POST format using EIGENVECTOR convention.
NONE	Default. No basis vectors will be printed.

Remarks:

1. **PERTURBATION** is an alternate format and is entirely equivalent to BASIS.
2. The post-processing file is named "*pname.DVG*".
3. See the POST command for the available formats.

7.7.4 DENSITY

Solution Control Entry: **DENSITY** - Topology Design Output Request

Description: Requests form of density output

Format:

$$\text{DENSITY} = \left\{ \begin{array}{l} \text{NONE} \\ \text{POST} \end{array} \right\}$$

Example:

DENS = POST

Option	Meaning
NONE	Default. No DENS density will be output.
POST	Density for all topology designed elements will be output to the DENS post processing file.

Remarks:

1. Normalized densities will be printed only for the topology designed elements.
2. The post-processing file is named “*pname*DENSxx.ext” and is created using the element strain energy analysis results style. See the POST command for the available formats.
3. The solution control command **OPRINT** controls which design cycles will write output into the DENSITY file.
4. The *pname*.DNS file is is not controlled by this command. See DOPT parameter **DNSHIS**.

7.7.5 DESIGN

Solution Control Entry: **DESIGN** - Design Output Request

Description: Requests printing of updated properties, materials and grid locations

Format:

$$\text{DESIGN} = \left\{ \begin{array}{c} \text{PROP} \\ \text{MAT} \\ \text{GRID} \\ \text{ALL} \\ \text{NONE} \end{array} \right\}$$

Examples:

DESIGN = GRID

DESIGN = BOTH

Option	Meaning
PROP	Updated properties will be printed.
MAT	Updated material properties will be printed.
GRID	Updated grid locations will be printed.
ALL	Both updated properties and grid locations will be printed.
NONE	Default. No design status will be printed.

Remarks:

1. If no DESIGN statement appears, updated properties, materials and grid locations will not be printed in the output file.
2. Regardless of the setting on the DESIGN statement in size/shape optimization, if the DOPT parameter **OPTHIS** is set to 1, updated properties and grid locations are written to the *pname*.OPT result file in a format suitable for inclusion in Bulk Data.

7.7.6 DPRINT

Solution Control Entry: **DPRINT** - Design Output Control

Description: Requests printed design results only at selected design cycles

Format:

$$\text{DPRINT} = \left\{ \begin{array}{c} \text{ALL} \\ \text{FIRST} \\ \text{LAST} \\ \text{FLAST} \\ n \\ \text{NONE} \end{array} \right\}$$

Examples:

DPRINT = ALL

DPRINT = 2

Option	Meaning
ALL	Default. Design results output at every Design Cycle.
FIRST	Design results output only at the first Design Cycle.
LAST	Design results output only at the last Design Cycle.
FLAST	Design results output only at the first and last Design Cycle.
n	Design results output for the initial, every n-th design cycle after the initial, and the final design cycle (Integer > 0).
NONE	No design results are printed.

Remarks:

1. If no DPRINT statement is provided in the solution control, all design results will be printed.

7.7.7 DRESP2

Solution Control Entry: **DRESP2** - Design Output Request

Description: Requests printing of synthetic response values created with DRESP2, DRESP3, TRESP2 and TRESP3 bulk data statements

Format:

$$\text{DRESP2} = \begin{Bmatrix} \text{YES} \\ \text{NO} \end{Bmatrix}$$

Examples:

DRESP2 = YES

DRESP2 = NO

Option	Meaning
YES	Default. Synthetic responses values will be printed
NO	No synthetic responses values will be printed.

7.7.8 DVBASIS

Solution Control Entry: **DVBASIS** - GRID Basis Vector Output Request

Description: Select and convert a set of GRIDs into DVGRID data (basis vectors) to be used in another input data. Also, create the associated DVAR data.

Format:

DVBASIS = dvid,lb,ub,setId

Examples:

DVBASIS = 10, -1.0, 1.0, 25

DVBASIS = 20, , , ALL

Option	Meaning
dvid	Design variable identification number (Integer>0).
lb	Lower bound of the design variable (real or blank). Default = -2.0.
ub	Upper bound of the design variable (real, ub>=lb). Default is 2.0.
setId	Set identification of a set of grids (Integer>0, ALL or blank). Default is ALL.

Remarks:

1. Multiple DVBASIS data are allowed in each input file.
2. This command creates **DVAR** and **DVGRID** data.
3. The DVAR and DVGRID data are stored in the *pname*.DVB file. The basis vectors can be used in another file using the bulk data command:
INCLUDE '*filename*.DVB'. Where *filename* is the project name of the file with the DVBASIS commands. Optionally, the user may cut and paste the results in the DVB files to its new input file.
4. The Solution Control command **DVGRID** = PRINT will print the DVGRID and DVAR data created by DVBASIS in the output file.
5. Two DVBASIS entries can not have the same design variable ID.

7.7.9 DVGRID

Solution Control Entry: **DVGRID** - Shape and Sizing Design Output Request

Description: Requests printed basis or perturbation vectors that were created by *GENESIS* using the DVGRIDC and DOMAIN data statements, the DTGRID data statement or the DVSHAPE or DVBASIS commands. The results are printed using DVGRID bulk data format in the output files

Format:

$$\text{DVGRID} = \begin{Bmatrix} \text{PRINT} \\ \text{NONE} \end{Bmatrix}$$

Examples:

DVGRID = PRINT

DVGRID = NONE

Option	Meaning
PRINT	DVGRID Output.
NONE	Default. Do not print DVGRID data created by DVGRIDC and DOMAIN or DVSHAPE or DVBASIS data in the output file.

Remarks:

1. Only the **DVGRID** data created by **DVGRIDC** + **DOMAIN**, **DVSHAPE**, **DVBASIS** and/or **DTGRID** data is printed.
2. DVGRID data created by DVSHAPE is always printed in the *pname.DVS* file, regardless of the setting of this entry.
3. DVGRID data created by DVBASIS is always printed in the *pname.DVB* file, regardless of the setting of this entry.

7.7.10 DVSHAPE

Solution Control Entry: **DVSHAPE** - Natural Basis Vector Output Request

Description: Select and convert a set of displacements into DVGRID data (natural basis or perturbation vectors) to be used in another input data. Also, create the associated DVAR data.

Format:

DVSHAPE = dvid,lb,ub,setid,maxp

Examples:

DVSHAPE = 10, -1.0, 1.0, 25, 10.0

DVSHAPE = 20, , , ALL, 1.0

DVSHAPE = 30, , , ALL

Option	Meaning
dvid	Design variable identification number (Integer>0).
lb	Lower bound of the design variable (real or blank). Default = -2.0.
ub	Upper bound of the design variable (real, ub>=lb). Default is 2.0.
setid	Set identification of a set of grids (Integer>0, ALL or blank). Default is ALL.
maxp	The displacements in setid are normalized so the maximum $\sqrt{U_x^2 + U_y^2 + U_z^2}$ is equal to MAXP (real or blank). If this field is left blank then no scaling is done and the displacements are translated directly to DVGRID data.

Remarks:

1. The DVSHAPE command can only be used in a static LOADCASE or LOADCOM.
2. Multiple DVSHAPE data are allowed in each loadcase.
3. This commands creates **DVAR** and **DVGRID** data.
4. The user must enter the **DOPT** parameter BASIS to indicate whether the generated DVGRID data should be in the basis or perturbation format.
5. The DVAR and DVGRID data are stored in the *pname.DVS* file. If the DOPT parameter is set to BASIS=0, then the DVGRID data contains perturbations. If the DOPT parameter is set to BASIS=1 then the DVGRID data contains basis vector components.
6. Two DVSHAPE entries can not have the same design variable ID.

7. The basis or perturbations vectors are to be used in another file that has the bulk data command: `INCLUDE 'filename.DVS'`. Where *filename* is the project name of the file with the DVSHAPE commands. Optionally, the user may cut and paste the results in the DVS files to its new input file.
8. The Solution Control command **DVGRID** = PRINT will print the DVGRID and DVAR data created by DVSHAPE in the output file.

7.7.11 GRAPH

Solution Control Entry: **GRAPH** - Design Output Request

Description: Requests the creation of a file with a graph of the objective function versus design cycle and a graph of the maximum constraint violation versus design cycle.

Format:

$$\text{GRAPH} = \begin{Bmatrix} \text{YES} \\ \text{NO} \end{Bmatrix}$$

Examples:

GRAPH = YES

GRAPH = NO

Option	Meaning
YES	Default. The graph file will be printed.
NO	The graph file will not be printed.

Remarks:

1. The DOPT parameter, **GRAPH**, controls the format of graph file. If the html format is selected, the graph file will contain a summary of the optimization results in addition to the graphs.

7.7.12 KAASENS

Solution Control Entry: **KAASENS** - Guyan Reduction Matrix Output Request

Description: Requests printing of the sensitivities of reduced stiffness matrix.

Format:

$$\mathbf{KAASENS} = \begin{Bmatrix} \text{POST} \\ \text{NONE} \end{Bmatrix}$$

Example:

KAASENS = POST

Option	Meaning
POST	Sensitivities of the reduced stiffness matrix will be output to the SKA post processing file.
NONE	Default. Do not print sensitivities of reduced stiffness matrix.

Remarks:

1. This command can only be used on reduced (ASET) eigenvalue loadcases.
2. The post processing file is named "*pnamexxy.SKA*" or "*pnamexxyyyyyyy.SKA*" where *pname* is the project name, xx corresponds to the design cycle number and yy or yyyyyyyy corresponds to the loadcase number, yy is used for the case where the loadcase id is less or equal to 99, yyyyyyyy is used for the case where the loadcase id is larger than 99.
3. The format of the ".SKA" file is described in [Sensitivities of Guyan Reduced Stiffness Matrix \(pnamexxy.SKA or pnamexxyyyyyyy.SKA\)](#) (p. 848).

7.7.13 MAASENS

Solution Control Entry: **MAASENS** - Guyan Reduction Matrix Output Request

Description: Requests printing of the sensitivities of reduced mass matrix.

Format:

$$\mathbf{MAASENS} = \begin{Bmatrix} \text{POST} \\ \text{NONE} \end{Bmatrix}$$

Example:

MAASENS = POST

Option	Meaning
POST	The sensitivities of reduced mass matrix will be printed to the SMA postprocessing file.
NONE	Default. Do not print sensitivities of reduced mass matrix.

Remarks:

1. This command can only be used on reduced (ASET) eigenvalue loadcases.
2. The post processing file is named “*pnamexxy.SMA*” or “*pnamexxyyyyyyy.SMA*” where *pname* is the project name, xx corresponds to the design cycle number and yy or yyyyyyyy corresponds to the loadcase number, yy is used for the case where the loadcase id is less or equal to 99, yyyyyyyy is used for the case where the loadcase id is larger than 99.
3. The format of the “.SMA” file is described in [Sensitivities of Guyan Reduced Mass Matrix \(pnamexxy.SMA or pnamexxyyyyyyy.SMA\)](#) (p. 849).

7.7.14 MODTRK

Solution Control Entry: **MODTRK**- Mode Tracking of Natural Frequencies or Buckling Load Factors Control

Description: Selects the use of mode tracking in an frequency and buckling load case

Format:

$$\text{MODTRK} = \begin{Bmatrix} \text{YES} \\ \text{ALL} \\ \text{NO} \end{Bmatrix}$$

Example:

MODTRK = YES

MODTRK = ALL

Option	Meaning
YES	The modes with constrained frequencies or buckling load factors are tracked
ALL	All modes are tracked
NO	Do not use mode tracking

Remarks:

1. This command takes precedence over the bulk data parameter MODTRK. The MODTRK parameter serves as a default for all eigenvalue loadcases without the solution control command MODTRK.

7.7.15 MPRINT

Solution Control Entry: **MPRINT** - Output Control

Description: Requests printed reduced matrices only at selected design cycles.

Format:

$$\text{MPRINT} = \left\{ \begin{array}{l} \text{ALL} \\ \text{FIRST} \\ \text{NONE} \end{array} \right\}$$

Example:

MPRINT = FIRST

Option	Meaning
ALL	Default. Print requested reduced matrices at every design cycle.
FIRST	Print reduce matrices only at the first design cycle.
NONE	Do not print reduced matrices.

Remarks:

1. This command is used to control the printing of the reduced stiffness and mass matrices and the sensitivity of the reduced stiffness and mass matrices at selected design cycles. These matrices are requested by the solution control commands KAA, MAA, **KAASENS** and **MAASENS**.

7.7.16 OPRINT

Solution Control Entry: **OPRINT** - Optimization Post-processing Output Control

Description: Requests results in optimization post-processing files only at selected design cycles

Format:

$$\text{OPRINT} = \left\{ \begin{array}{c} \text{ALL} \\ \text{FIRST} \\ \text{LAST} \\ \text{FLAST} \\ n \\ \text{NONE} \end{array} \right\}$$

Examples:

OPRINT = ALL

OPRINT = 2

Option	Meaning
ALL	Default. Design results output at every Design Cycle.
FIRST	Design results output only at the first Design Cycle.
LAST	Design results output only at the last Design Cycle.
FLAST	Design results output only at the first and last Design Cycle.
n	Design results output for the initial, every n-th design cycle after the initial, and the final design cycle (Integer > 0).
NONE	No design results are printed.

Remarks:

1. OPRINT affects optimization post-processing files controlled by the **DENSITY**, **SHAPE**, **SIZING** and **THICKNESS** solution control commands.

7.7.17 PERTURBATION

Solution Control Entry: **PERTURBATION** - Shape and Sizing Design Output Request

Description: Requests the printing in a post processing file the perturbation vector for each shape design variable.

Format:

$$\text{PERTURBATION} = \left\{ \begin{array}{l} \text{POST} \\ \text{NONE} \end{array} \right\}$$

Example:

PERTURBATION = POST

Option	Meaning
POST	Perturbation vectors are printed in POST format using eigenvector convention.
NONE	Default. No perturbation vectors will be printed.

Remarks:

1. BASIS is an alternate format and is entirely equivalent to PERTURBATION.
2. The post-processing file is named "*pname*.DVG".
3. See the POST command for the available formats.

7.7.18 RELIABILITY

Solution Control Entry: **RELIABILITY** - Reliability Output Request

Description: Requests that the reliability analysis results be printed in the output file at each design cycle of the optimization process

Format:

$$\text{RELIABILITY} = \left\{ \begin{array}{l} \text{ALL} \\ \text{USER} \\ \text{NONE} \end{array} \right\}$$

Examples:

RELIABILITY = ALL

RELIABILITY = NONE

Option	Meaning
ALL	The reliability analysis results of all retained constraints will be printed at each design cycle
USER	Default. The reliability analysis results of all retained constraints that have an allowable probability of failure value will be printed at each design cycle
NONE	No reliability analysis results will be printed.

Remarks:

1. This command will be used only if reliability analysis data is used. Reliability data is explained in the following section: **Random Variables for Reliability Analysis** (p. 267).
2. Probability of failure values are explained in the following section: **Allowable Probability of Failure on Constraint Entries** (p. 329).

7.7.19 SENSITIVITY

Solution Control Entry: **SENSITIVITY** - Output Control

Description: Requests printing of the sensitivity table

Format:

$$\text{SENSITIVITY} = \left\{ \begin{array}{c} \text{PRINT} \\ \text{POST} \\ \text{BOTH} \\ \text{NONE} \end{array} \right\}$$

Examples:

`SENSITIVITY = PRINT`

`SENSITIVITY = BOTH`

Option	Meaning
PRINT	Sensitivities of the retained responses with respect to the design variables will be printed in the output file.
POST	The sensitivity post processing file will be generated.
BOTH	Both printed and post processing output will be generated.
NONE	Default. No sensitivity data will be output.

Remarks:

1. The sensitivities of the retained constrained responses and objective function response with respect to the independent design variables are output.
2. The post processing file has the extension “.SEN”.
3. Sensitivity in topology optimization problems is not printed.

7.7.20 SHAPE

Solution Control Entry: **SHAPE** - Shape and Sizing Design Output Request

Description: Requests printing of shape changes during optimization to post processing file.

Format:

$$\text{SHAPE} = \left\{ \begin{array}{l} \text{POST} \\ \text{NONE} \end{array} \right\}$$

Examples:

SHAPE = POST

Option	Meaning
POST	The SHAPE post-processing file will be generated.
NONE	Default. No shape file will be generated.

Remarks:

1. The file is printed in a PUNCH, PATRAN or IDEAS file using the displacement specification. To animate the shape, use a post-processing program that supports transient analysis and PUNCH, PATRAN or IDEAS files.
2. The post-processing file is named "*pname*.SHP," for PUNCH or IDEAS format or "*pnamexxx*.SHP" for PATRAN format, where *pname* is the project name.
3. The solution control command **OPRINT** controls which design cycles will write output into the SHAPE file.
4. If mesh smoothing is used, the PSMOOTH parameter can be used to print the shape changes due to both mesh smoothing and shape optimization.

7.7.21 SIZING

Solution Control Entry: **SIZING** - Shape and Sizing Design Output Request

Description: Requests printing of element properties to post processing file.

Format:

$$\text{SIZING} = \begin{Bmatrix} \text{NONE} \\ \text{POST} \end{Bmatrix}$$

Examples:

SIZING = POST

Option	Meaning
NONE	Default. No element properties will be output.
POST	Element properties for all appropriate elements will be output to the OPOST post processing file.

Remarks:

1. The post-processing file is named "*pname*OPOSTxx.ext" and is created using the element strain energy analysis result style. Only POST formats PUNCH and OUTPUT2 are supported by this command.
2. The output will consist of records for each element property field containing data for all elements of the corresponding type. By default, output records will only be generated for a given property field if at least one property has design data associated to that property field. If the optimization parameter **OPOST** is set to 1, then all element property fields are output even if they are never designed.
3. The solution control command **OPRINT** controls which design cycles will write output into the OPOST file.

7.7.22 SSOL

Solution Control Entry: **SSOL** - Design Output Request.

Description: Requests the creation of the SSOL post processing file containing grid data and solid elements to reveal the thicknesses of shell and composite elements.

Format:

$$\text{SSOL} = \left\{ \begin{array}{c} \text{NO} \\ \text{YES} \\ \text{YES, FIXNORM} \end{array} \right\}$$

Examples:

SSOL = YES

SSOL = YES, FIXNORM

Option	Meaning
NO	Default. Do not create the SSOL results file.
YES	Create the SSOL results file.
YES, FIXNORM	Create the SSOL results file after adjusting shell and composite element connectivities such that adjacent elements have normals pointing in a consistent direction.

Remarks:

1. The SSOL results will be printed in a file named *pname*SSOLxx.dat. Where, *pname* is the *GENESIS* project name. In an optimization run, xx corresponds to the final design cycle. In an analysis run xx is 00.
2. The SSOL results file contains grid data and solid element connectivities (CHEXA and CPENTA elements) that reveal the thicknesses of CQUAD4 and/or CTRIA3 elements in the input data file. The 2-D elements are associated to “surfaces”, which are defined by groups of connected elements that refer to the same material, and such that the norms of adjacent elements are not too different. Each identified “surface” is represented by a PSOLID in the SSOL results file. If element norms are not consistent (which is acceptable for PSHELL or PCOMP with the SYM option, and no offsets on the CQUAD4/CTRIA3 entries), then the FIXNORM option can be used to correct the norms before applying the surface identification algorithm. No attempt is made to accurately model the intersections of different surfaces.

7.7.23 THICKNESS

Solution Control Entry: **THICKNESS** - Shape and Sizing Design Output Request

Description: Requests form of thickness output

Format:

$$\text{THICKNESS} = \begin{cases} \text{NONE} \\ \text{POST} \end{cases}$$

Examples:

THICK = POST

Option	Meaning
NONE	Default. No thicknesses will be output.
POST	Thickness for all appropriate elements will be output to the OPOST post processing file.

Remarks:

1. This command is similar to the **SIZING** command with the exception that output is restricted to thicknesses for CQUAD4 and CTRIA3 elements. For elements referencing PCOMP properties, the total thickness is output.
2. The post-processing file is named "*pname*OPOSTxx.ext" and is created using the element strain energy analysis result style. See the POST command for the available formats.
3. The solution control command **OPRINT** controls which design cycles will write output into the OPOST file.
4. The analysis parameter OPPTHK can be used to change element type indicated in the post processing file. This can be useful for visualizing the thickness file in conjunction with the SSOL file.

7.7.24 TSURF

Solution Control Entry: **TSURF** - Topology Design Output Request.

Description: Requests the creation of a post processing file containing isodensity surfaces. This command controls the density levels.

Format:

$$\text{TSURF} = \left\{ \begin{array}{c} \text{YES} \\ \text{NO} \\ \text{YES,L1,L2,...,L10} \end{array} \right\}$$

Example:

TSURF = YES, 45, 60, 75, 90

TSURF= NO

Option	Meaning
YES	Default. Create one isodensity surface using L1=35.
NO	Do not create isodensity surfaces file.
YES,L1,L2...,L10	Create up to ten isodensity surfaces using levels L1, L2,..., L10.

Remarks:

1. This command can be used in a topology optimization run or in a **TSURFACE** mode run. For more details on TSURFACE mode runs, see the executive control command TSURFACE.
2. All of the isodensity surfaces will be printed in a file named *pname*TSURFxx.dat. Where, *pname* is the *GENESIS* project name. In a topology optimization run, xx corresponds to the final design cycle. In a TSURFACE run, xx correspond to the design cycle specified on the TSURFACE executive control command.
3. The levels to be printed are problem dependent and the users are encouraged to run their problems with different density levels to find out the ones that work best for them. Here are some levels that have worked well for certain problems:
4. Shell models: Try L1 = 35, L2 = 50 and L3 = 75
5. Solid models: Try L1 = mass fraction percentage minus five

7.7.25 TVAR

Solution Control Entry: **TVAR** - Design Output Request

Description: Requests that design variables be printed in the output file at each design cycle and in a summary at the end of the optimization process

Format:

$$\text{TVAR} = \left\{ \begin{array}{c} \text{ALL} \\ \text{USER} \\ \text{NONE} \end{array} \right\}$$

Examples:

TVAR = ALL

TVAR = NONE

Option	Meaning
ALL	All design variables will be printed at each design cycle and a design variables summary will be printed at the end of the output file.
USER	Default. User defined design variables will be printed at each design cycle and a design variables summary will be printed at the end of the output file.
NONE	No design variables will be printed in the output file.

Remarks:

1. Topology and topometry design variables are printed with ID numbers that do not correspond to user element ID numbers.
2. The *pname.DNS* file is optionally created for topology optimization. This file contains the design variable values of each designed element at each design cycle according to the user element ID numbers. See [Topology Density File \(pname.DNS\)](#) (p. 842) for the format of the *pname.DNS* file. The creation of this file is controlled by the DOPT parameter [DNSHIS](#).
3. The *pname.HIS* file is always created. This file contains all the design variable values of each design cycle.

7.7.26 UPRINT

Solution Control Entry: **UPRINT** - Updated Model Output Control

Description: Requests printed an updated model only at selected design cycles

Format:

$$\text{UPRINT} = \left\{ \begin{array}{c} \text{ALL} \\ \text{FIRST} \\ \text{LAST} \\ \text{FLAST} \\ n \\ \text{NONE} \end{array} \right\}$$

Examples:

UPRINT = ALL

UPRINT = 2

Option	Meaning
ALL	Update input file is output at every Design Cycle.
FIRST	Update input file is output only at the first Design Cycle.
LAST	Update input file is output only at the last Design Cycle.
FLAST	Update input file is output only at the first and last Design Cycle.
n	Update input file is output for the initial, every n-th design cycle after the initial, and the final design cycle (Integer > 0).
NONE	Default. No update input file is printed.

Remarks:

1. If no UPRINT statement is provided in the solution control, no updated input file will be printed.
2. The update model will be printed in a file named *pname*UPDATExx.dat. Where, *pname* is the *GENESIS* project name and *xx* corresponds to the design cycle. This file is written using GENESIS input data format.
3. Only analysis data is printed. Design data is not printed in the “UPDATE” file.

CHAPTER 8

Shape, Sizing, Topography, Topometry and Freeform Design Model Data

- Shape and Sizing Design Data Relationships
- Shape, Sizing, Topography, Topometry and Freeform Design Bulk Data

8.1 Shape and Sizing Design Data Relationships

The chart below shows the basic relationships among the data statements for design using *GENESIS*. It includes all design commands that may be included in the input data file. The chart also includes the most relevant analysis data that are referenced by the design data.

This chart can be used to rapidly find the command names required to input a model for a particular design task. For example, if the user needs to create a design model for shape optimization, the chart shows that DVGRID data must reference DVAR data, as well as GRID analysis data. The DVGRID data may optionally be referenced by DOPT data.

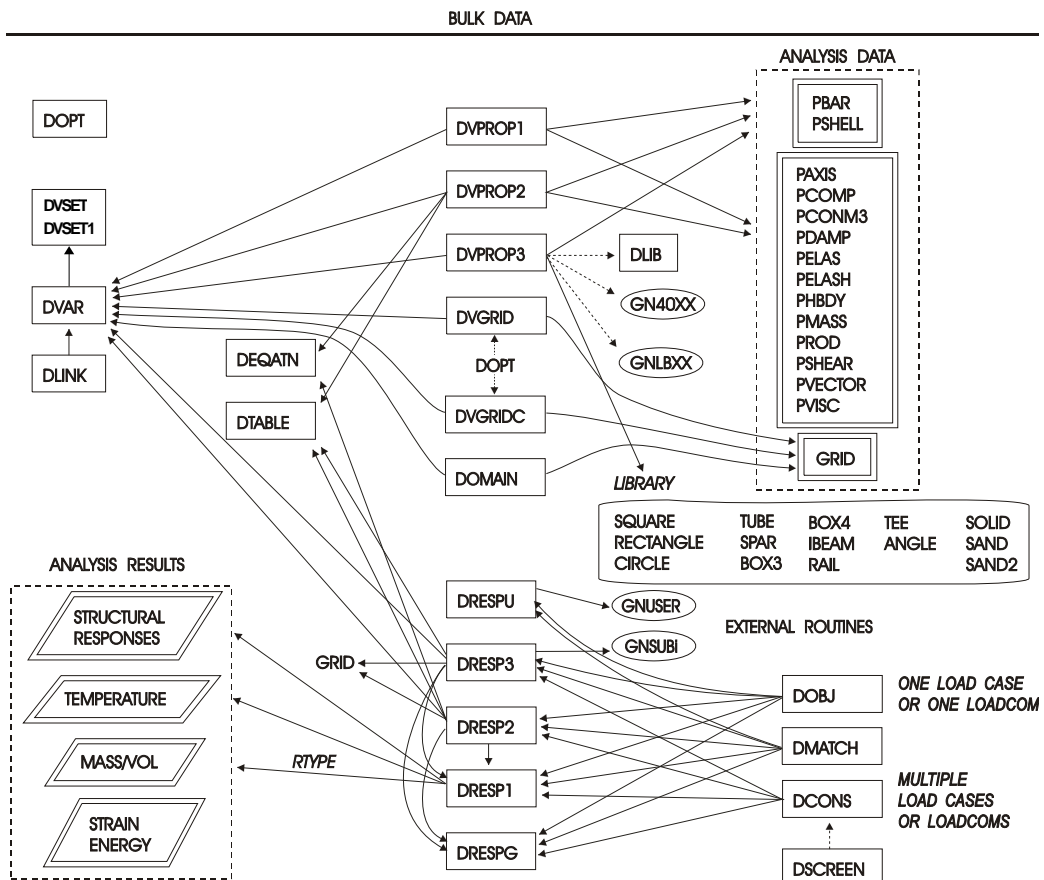


Figure 8-1

8.2 Shape, Sizing, Topography, Topometry and Freeform Design Bulk Data

The bulk data for design optimization using *GENESIS* is defined in this section. The data is given in alphabetical order.

8.2.1 DCONS

Data Entry: **DCONS** - Design Constraints.

Description: Define design constraints.

Format:

1	2	3	4	5	6	7	8	9	10
DCONS	RID	LID1	LB1	UB1	LID2	LB2	UB2	PF1	PF2
+		LID3	LB3	UB3	-etc.-				

Example:

1	2	3	4	5	6	7	8	9	10
DCONS	15	3	-20.0	20.0	4	-15.0	30.0		
+		6	-1.0	0.5					

Example of constraints with allowable probability of failure of 0.05 on loadcases 3,4 and 6:

1	2	3	4	5	6	7	8	9	10
DCONS	25	3	-20.0	20.0	4	-15.0	30.0	0.05	0.05
+		6	-1.0	0.5				0.05	

Alternate Formats:

1	2	3	4	5	6	7	8	9	10
DCONS	STRESS								
DCONS	GSTRESS								
DCONS	SURFACE								

Field Information Description

2	RID	DRESP1 , DRESP2 , DRESP3 or DRESPG entry identification number or one of the keywords STRESS, GSTRESS or SURFACE (Integer > 0 or "STRESS", "GSTRESS" or "SURFACE").
3,6	LIDi	Load case identification number (Integer ≥ 0 or blank, or LID1 may also be 'ALL'. Default = ALL). See Remarks 3-6.
4,7	LBi	Constraint lower bound imposed on this response quantity. (Real or blank. Default = -1.0E30).
5,8	UBi	Constraint upper bound imposed on this response quantity. (Real or blank, $LB \leq UB$. default = 1.0E30).

9,10	PFi	Allowable probability of failure. Failure means the response value is outside the bounds. (Real > 0.0 or blank. default = 1.0E30). See Remark 13.
------	-----	---

Remarks:

1. A DRESP1, DRESP2, DRESP3 or DRESPG entry can be referred to by at most one DCONS entry. A DCONS entry cannot reference a DRESP1, DRESP2, DRESP3 or DRESPG statement that is referred to by a [DCONS2](#), [DOBJ](#), [DINDEX](#), [DMATCH](#) or [DMATCH2](#) statement. However, two DRESP1, DRESP2, DRESP3 or DRESPG statements can be created with different IDs, but otherwise identical, with one being referred to by a DCONS2, DOBJ, DINDEX, DMATCH or DMATCH2 statement and the other by a DCONS statement.
2. The continuation data is optional.
3. If LID1 = ALL, LB1 and UB1 will apply to all load cases. In this case, no subsequent LIDs and LB, UB should be specified.
4. If a constraint references a MASS, VOLUME or INERTIA response, the load case identification number (LID) must be 0 or blank.
5. If a constraint references DRESP2 or DRESP3 responses that use the DRESP1L keyword, then the loadcase identification number (LID) must be blank.
6. Data for LID3, LID4, etc. may be specified on a continuation.
7. If a constraint references a DRESP2 or DRESP3 entry, which does not reference any DRESP1 entries, the load case identification number must be blank.
8. If LBi and UBi are specified for only some of the load cases, the response is constrained only in these specified load cases.
9. Lower bounds < -1.0E+29 and upper bounds > 1.0E+29 are ignored and will not generate constraints. If LBi or UBi are blank, then no constraint will be generated for that bound.
10. If a constraint points to a frequency response, the bounds imposed must be in Hz and only the lower or upper bound should be specified. To generate both a lower and upper bound frequency constraint, use two DRESP1 and two DCONS statements.
11. Automatic generation of static stress constraints can be requested using the RID=STRESS option. The format of this data is shown in the second example above. This option uses information on stress limits provided in MAT1, MAT2 and MAT8 data statements. For more detail on this option see [Automatic Generation of Element Stress Constraints](#) (p. 143).
12. Automatic generation of static grid stress constraints can be requested using the RID = GSTRESS or RID = SURFACE options. These options use the stress limits provided in the MAT1, MAT2 and MAT8 data statements. For more details on these options, see [Automatic Generation of Grid Stress Constraints](#) (p. 147).
13. The allowable probability of failure values are only used when random variables are defined in the problem. The allowable probability of failure values are only used in probabilistic design cycles. For more details, see [Reliability Based Optimization](#) (p. 191).

8.2.2 DCONS2

Data Entry: **DCONS2** - Design Constraints.

Description: Define design constraints using scale factors of responses' initial analysis values.

Format:

1	2	3	4	5	6	7	8	9	10
DCONS2	RID	LID1	LBF1	UBF1	LIDF2	LB2F	UBF2	PF1	PF2
+		LID3	LBF3	UBF3	-etc.-				

Example:

1	2	3	4	5	6	7	8	9	10
DCONS2	15	3	0.5	1.5.0	4	0.0	1.0.		
+		6		0.75					

Field Information Description

2	RID	DRESP1 , DRESP2 , DRESP3 or DRESPG entry identification number (Integer > 0).
3,6	LIDi	Load case identification number (Integer ≥ 0 or blank, or LID1 may also be 'ALL'. Default = ALL). See Remarks 3-6.
4,7	LBFi	Scale factor to calculate the constraint lower bound imposed on this response quantity. (Real or blank. Default = no constraint).
5,8	UBFi	Scale factor to calculate the constraint upper bound imposed on this response quantity. (Real or blank, $LBF \leq UBF$. default = no constraint).
9,10	PFi	Allowable probability of failure. Failure means the response value is outside the bounds. (Real > 0.0 or blank. default = 1.0E30). See Remark 13.

Remarks:

1. A DRESP1, DRESP2, DRESP3 or DRESPG entry can be referred to by at most one DCONS2 entry. A DCONS2 entry cannot reference a DRESP1, DRESP2, DRESP3 or DRESPG statement that is referred to by a **DCONS**, **DOBJ**, **DINDEX**, **DMATCH** or **DMATCH2** statement. However, two DRESP1, DRESP2, DRESP3 or DRESPG statements can be created with different IDs, but otherwise identical, with one being referred to by a DCONS, DOBJ, DINDEX, DMATCH or DMATCH2 statement and the other by a DCONS2 statement.
2. The continuation data is optional.

DCONS2 Shape, Sizing, Topography, Topometry and Freeform Design Model Data

3. If LID1 = ALL, LBF1 and UBF1 will apply to all load cases. In this case, no subsequent LIDs and LBF, UBF should be specified. However, the actual bound applied to each loadcase will dependent on the results of each loadcase.
4. If a constraint references a MASS, VOLUME or INERTIA response, the load case identification number (LID) must be 0 or blank.
5. If a constraint references DRESP2 or DRESP3 responses that use the DRESP1L keyword, then the loadcase identification number (LID) must be blank.
6. Data for LID3, LID4, etc. may be specified on a continuation.
7. If a constraint references a DRESP2 or DRESP3 entry, which does not reference any DRESP1 entries, the load case identification number must be blank.
8. If LBFi and UBFi are specified for only some of the load cases, the response is constrained only in these specified load cases.
9. If LBFi or UBFi are blank, then no constraint will be generated for that bound.
10. If a constraint points to a frequency response, only the lower or upper bound should be specified. To generate both a lower and upper bound frequency constraint, use two DRESP1 and two DCONS2 statements.
11. Additional information on DCONS2 can be found in the following chapter:
[Relative Constraint Bounds](#) (p. 328)
12. Scaling a negative number with a factor above 1.0 will result in a number less than that with a factor below 1.0. Therefore, care should be taken when using responses that may have negative values to insure that the calculated lower bound is less than the calculated upper bound. If you wish to set a scale factor bound on the magnitude of such a response, then it is recommended to use a DRESP3 with the NORM1 function.
13. The allowable probability of failure values are only used when random variables are defined in the problem. The allowable probably of failure values are only used in probabilistic design cycles. For more details, see [Reliability Based Optimization](#) (p. 191).

8.2.3 DEQATN

Data Entry: **DEQATN** - Equation Application.

Description: Define equation.

Format:

1	2	3	4	5	6	7	8	9	10
DEQATN	EQID	EQUATION							
+	EQUATION (Cont.)								

Example 1:

1	2	3	4	5	6	7	8	9	10
DEQATN	2	F1(A,B,C,D,R) = A+B*C-(D**3+10.0)+SIN(3.14159*R)							
+	+A**2/(B-C)								

Example 2:

1	2	3	4	5	6	7	8	9	10
DEQATN	3	F(A1, A2, A3, A4, A5, A6, A7, A8, A9, A10, A11, A12, A13, A14, A15, A16, A17, A18,							
+	A19, A20) = A1+A2+A3+A4+A5+A6+A7+A8+A9+A10+A11+A12+A13+A14+A15+A16+A17+								
+	A18+A19+A20								

Example 3: Layered equation.

1	2	3	4	5	6	7	8	9	10
DEQATN	3	F(A1, A2, A3, A4, A5, A6, A7, A8, A9, A10) = A1+A2+A3; G=F+A4+A5+A6 ;							
+	H = F/G+A7+A8+A9+A10								

Field Information Description

2	EQID	Unique equation identification number. (Integer > 0).
3-...	EQUATION	Equation. See remarks.

Remarks:

1. See [Equation Utility](#) (p. 271) for a discussion of the user defined Equation feature.
2. EQUATION consists of the collection of data in fields 3 through 10 on the first entry and fields 2 through 10 on the continuations. The boundaries between these fields are not recognized and the collection is treated as if it were one field.
3. EQUATION may contain embedded blanks.
4. The left-hand side of the first equation must include the complete list of input parameters, enclosed in parentheses, used in the right-hand side of the equations. It must not include the names of the lower level equations.

- The DEQATN entry is referenced by **DRESP2** or **DVPROP2** data entries. The parameters in the left-hand side of the equation correspond sequentially to NDVi, NCj, (NGn-NGCn), and (NRk-LIDRk) on DRESP2 entries. The parameters on the left hand side of the equation correspond sequentially to the DVIDi and CIDj fields on the DVPROP2 entries.
- The arithmetic operators in order of precedence are: ******, *****, **/**, **+**, **-**. The following relational operators may also be used: **=**, **/=**, **<**, **<=**, **>**, **>=**. The relational operators result in a value of 1.0 if the relation they are testing is true, or a value of 0.0 if the relation is false. The relational operators have lower precedence than the arithmetic operators.
- The following table lists the available intrinsic functions:

Function	Description
ABS(x)	Absolute value of x
ACOS(x)	Inverse cosine of x (result in radians)
ACOSH(x)	Inverse hyperbolic cosine of x
ASIN(x)	Inverse sine of x (result in radians)
ASINH(x)	Inverse hyperbolic sine of x
ATAN(x)	Inverse tangent of x (result in radians)
ATAN2(y,x)	Inverse tangent of y/x (result in radians, -π to π)
ATANH(x)	Inverse hyperbolic tangent of x
ATANH2(y,x)	Inverse hyperbolic tangent of y/x
AVG(x1,x2,...,xn)	Average: $(x1+x2+...+xn)/n$
COS(x)	Cosine of x (x in radians)
COSH(x)	Hyperbolic cosine of x
COTAN(x)	Cotangent of x (x in radians)
DIM(x,y)	Maximum of (0, x-y)
EXP(x)	e raised to power x
INT(x)	Convert x to integer
LOG(x)	Natural (base e) logarithm of x
LOG10(x)	Common (base 10) logarithm of x
LOGX(x,y)	Base x logarithm of y
MAX(x1,x2,...,xn)	Maximum of (x1, x2, ..., xn)
MIN(x1,x2,...,xn)	Minimum of (x1, x2, ..., xn)
MOD(x,y)	Remainder of x/y
PI(x)	π times x

Function	Description
RSS(x1,x2,...,xn)	Square root of sum of squares: SQRT(x1**2 + x2**2 + ... + xn**2)
SIGN(x,y)	Absolute value of x times sign of y
SIN(x)	Sine of x (x in radians)
SINH(x)	Hyperbolic sine of x
SQRT(x)	Square root of x
SSQ(x1,x2,...,xn)	Sum of squares: (x1**2 + x2**2 + ... + xn**2)
SUM(x1,x2,...,xn)	Sum: (x1 + x2 + ... + xn)
TAN(x)	Tangent of x (x in radians)
TANH(x)	Hyperbolic tangent of x

8. The relational operators and the functions ABS, DIM, INT, MAX, MIN, MOD, and SIGN should be used with caution, because they can create discontinuities in the function or its derivative. Such discontinuities can cause poor convergence behavior of the optimizer.
9. The maximum number of characters that can be used to define the function names and each of the arguments is 31.
10. Layered equations can be used by separating the equations with semi colons (;). The first equation contains the argument list for all the equations. Equations may reference the value of one or more preceding equations. The value of the last equation is the result of the DEQATN and is the value used by the DRESP2 or DVPROP2 entry.
11. Constants specified in **DTABLE** data can be used in DEQATN data without passing them through the argument list. If the constant name on DTABLE is the same as an argument list or function name, then the constant in DTABLE is not used in the equation.

8.2.4 DINDEX

Data Entry: **DINDEX** - Design Multi-objective Index Definition.

Description: Define the design objective index function.

Format:

1	2	3	4	5	6	7	8	9	10
DINDEX	RID1	LID1	W1	RID2	LID2	W2	RID3	LID3	W3
+	RID4	LID4	W4	-etc.-					

Example:

1	2	3	4	5	6	7	8	9	10
DINDEX	1	101	0.25						
+	2	102	0.50						
+	3	103	0.25						

Field Information Description

2, 5, 8	RIDi	Response entry (DRESP1 , DRESP2 , DRESP3 , DRESPG or DRESPU) ID. (Integer > 0).
3, 6, 9	LIDi	Load case identification number. (Integer > 0 or blank, Default = the first load case) If RID corresponds to mass, inertia or volume response, LID is ignored. Blank for DRESPU responses. See Remark 4.
4, 7, 10	Wi	Weighting factor. (Real ≠ 0). See Remark 7.

Remarks:

1. Only one DINDEX statement is allowed in the input data.
2. DRESP1, DRESP2 and DRESP3 entries referenced by the DINDEX data can define only a single response per load case.
3. Only one of **DOBJ** or DINDEX is allowed in the input data. DINDEX can be used with either **DMATCH** or **DMATCH2**.
4. If DINDEX references DRESP2 and/or DRESP3 responses that use the DRESP1L keyword, then the corresponding loadcase ID (LID) must be the same as the first LID specified in the DRESP1L data.
5. Fields 5-10 may be left blank.

6. If the DOPT parameter, DINDEXM, is 0 or 1, then responses are normalized by their values in the first design cycle. If DINDEXM is 2 or 3, then responses are not normalized.

- DINDEXM = 0 or 1

$$R_i = \frac{\text{Resp}_i}{|\text{Resp}_{0i}|}$$

- DINDEXM = 2 or 3

$$R_i = \text{Resp}_i$$

7. The multi-objective index function is calculated using the following equation:

$$T = \sum_{i=1} f_i$$

If the DOPT parameter DINDEXM is 0 or 2, then the objective function terms are calculated using reciprocals for negative weighting factors. If DINDEXM is 1 or 3, then the compliance index objective function is a straight summation.

- DINDEXM = 0 or 2

$$f_i = \begin{cases} W_i \cdot R_i & \text{if } W_i > 0 \\ \frac{-W_i}{R_i} & \text{if } W_i < 0 \end{cases}$$

- DINDEXM = 1 or 3

$$f_i = W_i R_i$$

Note that in all cases, using a positive weighting factor will drive the corresponding response to be minimized, while a negative weighting factor will drive the corresponding response to be maximized.

8. Typical weighting factor usage;

Response	W_i Sign
FREQ	NEGATIVE (to maximize)
MASS SENERGY	POSITIVE (to minimize)

9. The multi-objective index function is always minimized. To maximize individual responses, use negative weighting factors.

8.2.5 DLIB

Data Entry: **DLIB** - User Defined Library Element Information.

Description: Define the information necessary to access a user defined element which has been linked to *GENESIS*.

Format:

1	2	3	4	5	6	7	8	9	10
DLIB	LIBID	TYPE	NCSD	NSP	NSPS	NSTR			
+	SP1	SP2	SP3	SP4	SP5	SP6			

Example:

1	2	3	4	5	6	7	8	9	10
DLIB	25	BEAM	4	3	5	8			
+	1	2	3	7	8				

Field Information Description

2	LIBID	Element library number (Integer, $15 \leq \text{LIBID} \leq 25$).
3	TYPE	PBAR or PSHELL.
4	NCSD	Number of cross-sectional dimensions (Integer ≥ 1).
5	NSP	Number of section properties, neglecting shear deformation ($1 \leq \text{Integer} \leq 5$ for PBAR or $1 \leq \text{Integer} \leq 2$ for PSHELL).
6	NSPS	Number of section properties, including shear deformation ($\text{NSP} \leq \text{Integer} \leq 6$ for PBAR or $\text{NSP} \leq \text{Integer} \leq 3$ for PSHELL).
7	NSTR	Number of stresses per element ($0 \leq \text{Even Integer} \leq 16$).
2-7	SPi	Section property codes (Integer or blank).

Remarks:

1. The SPi must be listed in increasing numerical order with no embedded blanks.

Section Properties:

CODE	PROPERTY
1	Bar element area.
2	I2 area moment of inertia (Iyy).
3	I1 area moment of inertia (Izz).
4	I - use if both I1 and I2 are the same.
5	I12 area moment of inertia.
6	J torsional moment of inertia.
7	AS1 shear area for plane 1 (ASy).
8	AS2 shear area for plane 2 (ASz).
9	Shear area - use if both AS1 and AS2 are the same.
100	PSHELL plate thickness (T).
101	Plate bending stiffness (D).
102	Plate shear thickness (TS)

2. If property code 5 (I_{12}) is calculated, then shear deformation must be excluded [i.e., property codes 7, 8 and 9 (shear areas) must not be calculated].
3. The **DLIB** executive control command should be used to identify the shared object (DLL) that contains the user-routines.
4. User defined elements are described in **User-Supplied Section Property Calculation Interface Function (DLIBPROP)** (p. 288). This section explains how to create the DLIBPROP interface function to define element properties as functions of design variables and how to create the DLIBSTRESS interface function (p. 292) to define stresses as a function of design variables and forces.

8.2.6 DLINK

Data Entry: **DLINK** - Multiple Design Variable Linking.

Description: Relate one design variable to one or more other design variables.

Format:

1	2	3	4	5	6	7	8	9	10
DLINK	DVID	C0	CMULT	DV1	C1	DV2	C2	DV3	C3
+	DV4	C4	-etc.-						

Example:

1	2	3	4	5	6	7	8	9	10
DLINK	15	0.0	3.0	20	5.2	13	1.0	14	2.0
+	8	1.0							

Field Information Description

2	DVID	Dependent DVAR identification number. (Integer > 0).
3	C0	Constant term. (Real or blank. Default = 0.0).
4	CMULT	Constant multiplier. (Real ≠ 0.0 or blank. Default = 1.0).
5,7,..	DVi	Independent DVAR identification number. (Integer > 0).
6,8,..	Ci	Coefficient i (corresponding to DV _i). (Real).

Remarks:

1. This entry specifies the relationship $DVID = C_0 + CMULT \sum_i C_i DV_i$
2. This capability provides a means of linking physical design variables such as element thicknesses to non-physical design variables such as the coefficients of shape functions.
3. CMULT provides a simple means of imposing a linear relationship amount to C_i . For example if $C_i = 1/7, 2/7, 4/7$, etc. is desired, then $CMULT = 0.142857$ ($CMULT \approx 1/7$) and $C_i = 1.0, 2.0, 4.0$, etc. may be input.
4. The same independent design variable, DV_i , must not occur on a DLINK entry more than once.
5. The design variable specified by DVID must be defined by a DVAR data statement.
6. Design variables made dependent by DLINK may not be used as independent on any DLINK entry.
7. Design variables made dependent by DLINK cannot be made discrete (DVSID field on DVAR must be blank).

8.2.7 DMATCH

Data Entry: **DMATCH** - Analysis Matching Objective Function.

Description: Defines the objective function to match analysis results with specified target values without weighting factors.

Format:

1	2	3	4	5	6	7	8	9	10
DMATCH	RID1	LID1	T1	R1	LID2	T2	RID3	LID3	T3
+	RID4	LID4	T4	-etc.-					

Example:

1	2	3	4	5	6	7	8	9	10
DMATCH	4	1	0.3	5	2	70.0			

Field Information Description

2, 5, 8	RIDi	Response entry (DRESP1 , DRESP2 , DRESP3 , DRESPG or DRESPU) ID. This identifies the response that is to be matched. (Integer > 0).
3, 6, 9	LIDi	Load case identification number. (Integer > 0 or blank, Default = the first load case) For DRESPG, MASS, VOLUME or INERTIA responses, LID is ignored. Blank for DRESPU responses. See Remark 4.
4, 7, 10	Ti	Target value for the response (Real).

Remarks:

1. DRESP1, DRESP2 and DRESP3 entries referenced by the DMATCH data can define only a single response per load case.
2. Only one of DMATCH, **DMATCH2** or **DOBJ** is allowed in the input. **DINDEX** may be used with DMATCH.
3. The DRESPU entry referred to by DMATCH also produces constraints on the objective function. Therefore, large lower (-1.0E30) and upper (1.0E30) bounds should be used to eliminate the constraints.
4. If the DMATCH references DRESP2 and/or DRESP3 responses that use the DRESP1L keyword, then the corresponding loadcase ID (LID) must be the same as the first LID specified in the DRESP1L data.
5. Fields 5-10 may be left blank.
6. To select the method to do the matching, use the DOPT parameter IMATCH. Two methods are available: the "Beta" method (IMATCH = 1) and the least squares method (IMATCH = 0). IMATCH = 0 is the Default. For more details on these two methods and their associated equations see **Matching Analysis Results** (p. 23).

DMATCH

Shape, Sizing, Topography, Topometry and Freeform Design Model Data

7. DMATCH is equivalent to DMATCH2 with all weighting factors (W_i) equal to 1.0.

8.2.8 DMATCH2

Data Entry: **DMATCH2** - Analysis Matching Objective Function (Alternate Form).

Description: Defines the objective function to match analysis results with specified target values and weighting factors.

Format:

1	2	3	4	5	6	7	8	9	10
DMATCH2	RID1	LID1	T1	W1	RID2	LID2	T2	W2	
+	RID3	LID3	T3	W3	-etc.-				

Example:

1	2	3	4	5	6	7	8	9	10
DMATCH2	4	1	25.0	2.0					
	5	1	35.0	1.0					

Field Information Description

2, 6	RIDi	Response entry (DRESP1 , DRESP2 , DRESP3 , DRESPG or DRESPU) ID. This identifies the response that is to be matched. (Integer > 0).
3, 7	LIDi	Load case identification number. (Integer > 0 or blank, Default = the first load case) For DRESPG , MASS , VOLUME or INERTIA responses, LID is ignored. Blank for DRESPU responses. See Remark 4.
4, 8	Ti	Target value for the response (Real).
5,9	Wi	Weighting factor (Real \neq 0.0 or blank, Default = 1.0). See Remark 7.

Remarks:

1. **DRESP1**, **DRESP2** and **DRESP3** entries referenced by the **DMATCH2** data can define only a single response per load case.
2. Only one of **DMATCH**, **DMATCH2** or **DOBJ** is allowed in the input. **DINDEX** may be used with **DMATCH2**.
3. The **DRESPU** entry referenced to by **DMATCH2** also produces constraints on the objective function. Large lower (-1.0E30) and upper (1.0E30) bounds could be used to eliminate these added constraints, if they are not needed.
4. If the **DMATCH2** references **DRESP2** and/or **DRESP3** responses that use the **DRESP1L** keyword, then the corresponding loadcase ID (LID) must be the same as the first LID specified in the **DRESP1L** data.
5. Fields 6-9 may be left blank.

DMATCH2

Shape, Sizing, Topography, Topometry and Freeform Design Model

6. To select the method to do the matching, use the DOPT parameter IMATCH. Two methods are available: the “Beta” method ($IMATCH = 1$) and the least squares method ($IMATCH = 0$). $IMATCH = 0$ is the Default. For more details on these two methods and their associated equations see [Matching Analysis Results](#) (p. 23).
7. The **RMATCH** DOPT parameter is used to refine the multipliers used in the calculation of the matching optimization equations.

8.2.9 DOBJ

Data Entry: **DOBJ** - Design Objective.

Description: Define a single design objective function.

Format:

1	2	3	4	5	6	7	8	9	10
DOBJ	RID	LABEL	LID	MIN/MAX					

Example:

1	2	3	4	5	6	7	8	9	10
DOBJ	4	VOLUME		MIN					

Field Information Description

2	RID	Response entry (DRESP1 , DRESP2 , DRESP3 , DRESPG or DRESPU) ID. This identifies the response that is to be the objective function. (Integer > 0).
3	LABEL	User defined name for output purposes (Characters or blank).
4	LID	Load case identification number. (Integer > 0 or blank, Default = the first load case) If RID corresponds to mass, inertia or volume response, LID is ignored. Blank for DRESPU responses. See Remark 4.
5	MIN/MAX	Defines minimization (MIN) or maximization (MAX) to be performed. (Default = MIN).

Remarks:

1. The DRESP1, DRESP2 or DRESP3 entry referenced by the DOBJ data can define only a single response per load case.
2. DOBJ cannot be used with any of **DINDEX**, **DMATCH** or **DMATCH2**.
3. The DRESPU entry referred to by DOBJ also produces constraints on the objective function. Therefore, large lower (-1.0E30) and upper (1.0E30) bounds should be used to eliminate the constraints.
4. If the objective function references a DRESP2 or DRESP3 response that uses the DRESP1L keyword, then the loadcase ID (LID) must be the same as the first LID specified in the DRESP1L data.

8.2.10 DOMAIN

Data Entry: **DOMAIN** - DOMAIN elements to be used with DVGRIDC data to automatically create basis or perturbation vectors (DVGRID data).

Description: Define a domain element. The domain element has corners and internal grids. The corners grids defines the basic shape of the domain element. If the user, using DVGRIDC data, define perturbations (basis) at the corner grids and/or at internal grids that are close to the middle of edges then the program automatically generates the perturbation (basis) at all grids in the DOMAIN using interpolation functions.

Format:

	1	2	3	4	5	6	7	8	9	10
DOMAIN	ID									
+	TYPE	CG1	CG2	CG3	CG4	CG5	CG6	CG7	CG8	
+	"DGRID"	G1	G2	G3	G4	G5	G6	G7	G8	
+		G9	G10	-etc.-						
+	"DVAR"	DV1	DV2	DV3	DV4	DV5	DV6	DV7	DV8	
+		DV9	DV10	-etc.-						

Alternate Format:

	1	2	3	4	5	6	7	8	9	10
DOMAIN	ID									
+	TYPE	CG1	CG2	CG3	CG4	CG5	CG6	CG7	CG8	
+	"GSET"	SETID								
+	"DVAR"	DV1	DV2	DV3	DV4	DV5	DV6	DV7	DV8	
+		DV9	DV10	-etc.-						

Example 1:

	1	2	3	4	5	6	7	8	9	10
DOMAIN	2									
+	TRIA3	1	3	6						
+	DGRID	15	16	17	18	19	20	21	22	
+		23	24	25						

Example 2:

	1	2	3	4	5	6	7	8	9	10
DOMAIN		3								
+		HEXA	201	301	202	402	501	601	502	602
+		GSET	3							
+		DVAR	1	7						

Field Information Description

2	ID	Unique Domain Identification number. (integer > 0).
2	TYPE	Domain element type: RBE2, BAR, TRIA3, QUAD4, TETRA, PENTA, HEXA, BARX, TRIAX3, QUADX4.
3, 4, ...	CGi	Corner Grid Ids (Integer >0). See remark 1.
2	DGRID	Word indicating grid numbers will follow.
3, 4, ...	Gi	Interior Grid Ids (Integer >0 or blank). See remark 2.
2	GSET	Word indicating grid set number will follow.
3	SETID	Set ID of a set defined by a Solution Control SET entry (Integer > 0).
2	DVAR	Word indicating design variable numbers will follow. See Remark 11.
3, 4, ...	DVi	DVAR entry identification number (Integer > 0).

DOMAIN

Shape, Sizing, Topography, Topometry and Freeform Design Model Data

Remarks:

1. The following table show the required number of corner grids for each type of domain element:

ELEMENT	GRIDS
RBE2	1
BAR	2
TRIA3	3
QUAD4	4
TETRA	4
PENTA	6
HEXA	8
BARX	2+2
TRIAX3	3+2
QUADX4	4+2

The corner grid list of the axisymmetric domains, BARX, TRIAX3 and QUADX4, contains the element corner grids followed by two grids that define the axis of axisymmetry.

2. There is no limit in the number interior grids or in the number of continuation lines.
3. The DOMAIN data is used with **DVGRIDC** data to create basis or perturbation vectors. To output the results of the generated vectors use **DVGRID = PRINT** in solution control. The output contains the basis or perturbation vectors using DVGRID format.
4. If there is no DVGRIDC data that references a grid of the DOMAIN then this DOMAIN will be ignored and a warning message will be issued.
5. The DOMAIN element has no structural meaning. A collection of DOMAIN elements can be thought of as an auxiliary model to construct basis or perturbation vectors.
6. When the perturbation (basis) defined in DVGRIDC data references an interior grid that is close to the middle edge of a DOMAIN element, the DOMAIN element includes that grid to do the interpolation. In other words, the DOMAIN elements include more interpolation functions than the basic ones implied by the number of the corner nodes. For example, a QUAD4 DOMAIN element has internally 4 to 8 interpolations functions associated to it.
7. A perturbation, defined with DVGRIDC data, of the corner grid of a RBE2 DOMAIN element will create the same perturbation to all interior grids of the RBE2 DOMAIN.

8. The following figures show the basic shape of the DOMAIN elements.

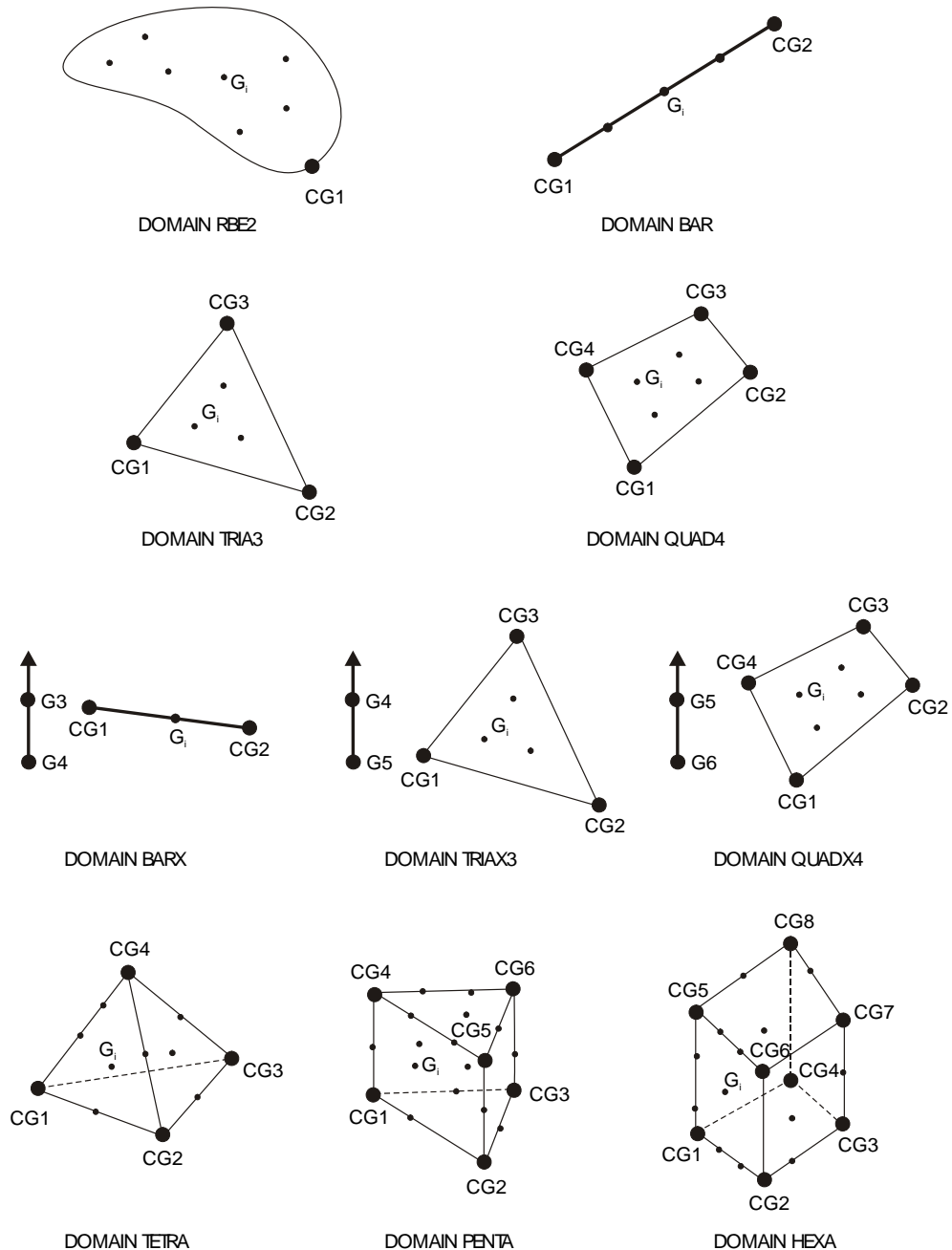


Figure 8-2

9. Internal nodes do not necessarily have to be inside the DOMAIN element. However, it is not recommended to have internal nodes outside the DOMAIN because that can cause numerical difficulties.

DOMAIN

Shape, Sizing, Topography, Topometry and Freeform Design Model Data

10. The optimization parameter DVGTOLE can be changed on the DOPT entry to allow a perturbation to be applied to a midside node far from the midpoint of an edge. This change has to be done with care to avoid numerical difficulties.
11. If the DVAR list is omitted, all appropriate shape design variables will be used with this domain.

8.2.11 DOPT

Data Entry: **DOPT** - Optimization parameters.

Description: Define parameters to be used in optimization.

Format:

1	2	3	4	5	6	7	8	9	10
DOPT	DESMAX								
+	NAM1	VAL1	NAM2	VAL2	NAM3	VAL3	NAM4	VAL4	
+	NAM5	VAL5	-etc.-						

Example:

1	2	3	4	5	6	7	8	9	10
DOPT	10								
+	CONV1	0.0001	IPRINT	3	CONVDV	0.01			

Field	Information	Description
-------	-------------	-------------

2	DESMAX	Maximum number of approximate optimizations to be performed. (Integer > 0 or blank, Default = 10).
2,4,...	NAMi	Name of additional parameter (see table below for available parameters).
3,5,...	VALi	Value of additional parameter. Real or Integer as indicated by the table below.

Remarks:

1. Only one DOPT statement may appear in the input data file.
2. DOPT is also used for topology optimization. See **DOPT** (p. 677).

Available shape and sizing optimization parameters are listed below.

Method Switches

PARAMETER	POSSIBLE VALUES	DEFAULT	DEFINITION
ADJOIN	-1 0 1	-1	Switch to select the sensitivity calculation method. If ADJOIN = 1, <i>GENESIS</i> will use the adjoint method for response sensitivities. If ADJOIN = 0, <i>GENESIS</i> will use the direct method. The adjoint method is generally faster for problems with many design variables and few constraints. If ADJOIN = -1 (the default), <i>GENESIS</i> will automatically select direct or adjoint.
BASIS	0 1	No Default	The Basis/Perturbation vector switch for shape design. For basis, use BASIS = 1. For perturbation, use BASIS = 0. See Basic Shape Design Capabilities (p. 72) for a detailed discussion.
ICOMPAPP	0 1	0	Switch for composite element approximation. If ICOMPAPP=1, the program will use special approximations that can use a large amount of disk space in some cases. If ICOMPAPP=0, the program will use direct approximations and usually will use less disk space.
DINDEXM	0 1 2 3	0	Design multi-objective index formulation. A value of 0 will cause <i>GENESIS</i> to use the reciprocal contribution for responses with negative weighting factors. Responses are normalized. A value of 1 will cause <i>GENESIS</i> to use the direct contribution for responses with negative weighting factors. Responses are normalized. A value of 2 will cause <i>GENESIS</i> to use the reciprocal contribution for responses with negative weighting factors. A value of 3 will cause <i>GENESIS</i> to use the direct contribution for responses with negative weighting factors. If shape/sizing data is mixed with topology data, DINDEXM must have the same value as TINDEXM .
DR1MV	0 1	1	Switch to control the way multiple property IDs are treated in a MASS or VOLUME response selection in DRESP1. A value of 0 will cause <i>GENESIS</i> to generate one mass or volume response per PID listed in the DRESP1 (i.e., generate a vector response). A value of 1 will cause <i>GENESIS</i> to sum the masses or volumes of all PIDs listed in the DRESP1. This causes <i>GENESIS</i> to generate a single response value for the DRESP1. This option is necessary when the DRESP1 is used in DOBJ, DINDEX, DMATCH or DMATCH2.

ISHLAPP	0 1	1 (0 with DSPLIT)	Switch for shell element approximation and DVPROP3. If ISHLAPP=1, the program will use 2 intermediate design variable per PSHELL+DVPROP3 property. If ISHLAPP=0, the default for topometry optimization, the program will use direct approximations and usually will use half the disk space for the sensitivity matrix.
LAMASNS	0 1 2	0	Controls the type of approximation used in calculating the sensitivity of the bucking load factors. A value of 0, the default, causes GENESIS to calculate the derivative exactly. A value of 1 will ignore the terms associated to displacement sensitivities. A value of 2 will ignore all sensitivities associated to the geometric stiffness matrix. This parameter should be use only as the last resort to solve problems where exact calculations are too expensive.
LINAPR	0 1 -1	0	Approximation Optimization Method Control LINAPR = 0 will cause the program to use the “fast” linear approximations method when all approximations are linear. If any approximations are nonlinear, then the program will use its standard hybrid approximations. LINAPR = 1 will force the program to use the “fast” linear approximations method for all responses. This will result in faster times in each design cycle, but it might lower the quality of approximations which in some cases might cause the program to need more design cycles to converge. LINAPR = -1 will cause to always use its standard linear and hybrid approximations.
MODAPP	0 1	0	Controls the type of approximation used in calculating the MDISP, MVELO and MACCE response types of DRESP1. If MODAPP=0, the default, the program will use direct approximations. If MODAPP=1, the program will use a special approximation better suited to capturing resonance. For most problems, the direct method is much faster and requires less memory.
IUGRAD	0 1	0	Switch for finite difference calculation of gradients of responses calculated by the user program (DRESPU). If IUGRAD=1, the user program will calculate gradients.
METHOD	1 2 3	1	Optimization method to be used by DOT. METHOD = 1 means use the modified method of feasible directions. METHOD = 2 means use sequential linear programming. METHOD = 3 means use sequential quadratic programming. If the problem is unconstrained (NCON=0), the BFGS algorithm will be used if METHOD=1 and the Fletcher-Reeves algorithm will be used if METHOD=2.
OPTM	0 1	0	Switch to select optimizer. If OPTM=0, GENESIS will use the DOT optimizer. If OPTM=1, GENESIS will use the BIGDOT optimizer. (Default=0 for Shape and Sizing and 1 for Topology).

RSHLAPP	0 1	0 (1 with STRDOT optimizer)	Switch for shell element forces approximations. If RSHLAPP=0, the program will use the standard force approximations. If RSHLAPP=1, the program will use a special fast approximation. The default is 0 except when the STRDOT optimizer is used in which case the default is 1. The standard force approximation is more accurate but uses more computation resources (time/disk and memory).
RCOMPAPP	0 1	0 (1 with STRDOT optimizer)	Switch for composite element forces approximations. If RCOMPAPP=0 the program will use the standard force approximations. If RCOMPAPP=1 the program will use a special fast approximation. The default is 0 except when the STRDOT optimizer is used in which case the default is 1. The standard force approximation is more accurate but uses more computation resources (time/disk and memory).

Continuous Variables Move Limit Parameters

PARAMETER	POSSIBLE VALUES	DEFAULT	DEFINITION
DELP	Real > 0.0	0.5	Fractional change allowed for each designed property during the approximate optimization. This provides move limits.
DPMIN	Real > 0.0	0.1	Minimum move limit factor imposed for properties. The minimum move limit is defined as: $\text{Minimum Move Limit} = \begin{cases} \text{DPMIN} & \text{if } P_{\text{initial}} \leq 1.0 \\ \text{DPMIN} P_{\text{initial}} & \text{if } P_{\text{initial}} > 1.0 \end{cases}$
DELX	Real > 0.0	0.5	Fractional change allowed for each design variable during the approximate optimization. This provides move limits.
DXMIN	Real > 0.0	0.1	Minimum move limit factor imposed for design variables. The minimum move limit is defined as: $\text{MinimumMoveLimit} = \begin{cases} \text{DXMIN} & \text{if } X_{\text{initial}} \leq 1.0 \\ \text{DXMIN} X_{\text{initial}} & \text{if } X_{\text{initial}} > 1.0 \end{cases}$

Move Limits Parameters for Topometry Optimization

PARAMETER	POSSIBLE VALUES	DEFAULT	DEFINITION
DXFRAC	Real > 0.0	1.0	The move limit parameters used by the design variables associated to topometry regions are multiplied by this parameter.

Parameters for Shape, Topography and Freeform Optimization

PARAMETER	POSSIBLE VALUES	DEFAULT	DEFINITION
SHAPECN	0 1 2 3	2	Distortion Constraint Controls SHAPECN = 0 will cause the program to not use distortion constraints. SHAPECN = 1 will cause the program to impose internal constraints to avoid that the jacobian of any designable element be negative. SHAPECN = 2 is similar to 1, but in addition it will impose constraints so that any distortion measure, as defined in DISTORT analysis entry, does not produce errors. If the user does not define DISTORT values, the program will use the DISTORT default values. SHAPECN = 3 is similar to SHAPECN = 2, but instead of using ERROR limits the program will use WARNING limits.
SHAPEGL	1 2 3	3	Location where distortion constraints are imposed. SHAPEGL = 1 will cause the program to impose internal constraints outside the approximate problem. This option can be used when the approximate problem takes significant time. SHAPEGL = 2 will cause the program to impose internal constraints in the approximate problem. SHAPEGL = 3 will cause the program to impose internal constraints in the approximate problem and outside of it.

Parameters for Hard Convergence

PARAMETER	POSSIBLE VALUES	DEFAULT	DEFINITION
CONV1	Real > 0.0	0.001	Relative change criterion to detect hard convergence of the overall optimization process. Terminate if the <u>relative</u> change in the <u>objective function</u> is less than CONV1 for two consecutive design cycles and all constraints are satisfied within a tolerance of GMAX.
CONV2	Real > 0.0	0.001 (0.000001 with DMATCH or DMATCH2)	Absolute change criterion to detect hard convergence of the overall optimization process. Terminate if the <u>absolute</u> change in the <u>objective function</u> is less than CONV2 for two consecutive design cycles and all constraints are satisfied within a tolerance of GMAX. $CONV2 = \text{Max}(CONV2 * OBJ_{initial} , 1.0E-19)$.
GMAX	Real > 0.0	0.005	Maximum <u>constraint</u> violation allowed at the optimum. Constraints are normalized so a value of 0.01 represents a one percent constraint violation, which is normally considered acceptable.

Parameters for Soft Convergence

PARAMETER	POSSIBLE VALUES	DEFAULT	DEFINITION
CONVCN	Real > 0.0	0.001	Allowable change in the maximum <u>constraint</u> value for soft convergence. If the change in the maximum constraint value is less than CONVCN, and CONVDV, CONVPR and CONVLC are satisfied, then terminate the design process with soft convergence.
CONVDV	Real > 0.0	0.001 (0.000001 with DMATCH)	Relative change criterion to detect soft convergence of the overall optimization process. Terminate with soft convergence if the maximum relative change in the <u>design variables</u> is less than CONVDV during the approximate optimization and CONVPR, CONVCN, and CONVLC are satisfied.
CONVLC	Real > 0.0	0.001	Maximum allowable grid location movement for soft convergence. Terminate with soft convergence if the maximum <u>grid location</u> movement is less than CONVLC, and CONVDV, CONVPR and CONVCN are satisfied.
CONVPR	Real > 0.0	0.001	Relative change criterion to detect soft convergence of the overall optimization process. Terminate if the maximum relative change in the <u>properties</u> is less than CONVPR during the approximate optimization and CONVDV, CONVCN, and CONVLC are satisfied.

Finite Difference Parameters for DRESPU responses

PARAMETER	POSSIBLE VALUES	DEFAULT	DEFINITION
FDCHMU	Real > 0.0	0.000001	Minimum step size for sensitivity (gradient) calculations for the user supplied subroutine (DRESPU data).
FDCHU	Real > 0.0	0.00001	Relative step size for sensitivity (gradient) calculations for the user supplied subroutine (DRESPU data).

Checking Parameters

PARAMETER	POSSIBLE VALUES	DEFAULT	DEFINITION
DVTOL	Real > 0.0	1.0E35	If the difference between the initial dependent variable and its input value is greater than DVTOL, a fatal error is generated.
DVGTOL	Real > 0.0	0.1666667	Parameter used to define if a grid in a DOMAIN referenced by a DVGRIDC is close or not to the midside of the DOMAIN.
PTOL	Real > 0.0	1.0E35	If the difference between the initial property and its input value is greater than PTOL, a fatal error is generated.
SGENEL	0 1	0	Switch to force <i>GENESIS</i> to ignore when a grid referenced by GENEL is being designed. A value of 0 will make <i>GENESIS</i> stop in case a grid referenced by GENEL is being designed. A value of 1 will make the program allow the design.
SK2UU	0 1	0	Switch to force <i>GENESIS</i> to ignore when a grid referenced by K2UU is being designed. A value of 0 will make <i>GENESIS</i> stop in case a grid referenced by K2UU is being designed. A value of 1 will make the program allow the design.
SM2UU	0 1	0	Switch to force <i>GENESIS</i> to ignore when a grid referenced by M2UU is being designed. A value of 0 will make <i>GENESIS</i> stop in case a grid referenced by M2UU is being designed. A value of 1 will make the program allow the design.
IZGRAD	0 1 2	0	A value of 0 will cause <i>GENESIS</i> to use zero gradient constraints in the same way as regular constraints. A value of 1 will cause <i>GENESIS</i> to ignore constraints that have zero gradients and that are violated. A value of 2 will cause <i>GENESIS</i> to ignore constraints that have zero gradients regardless if they are violated or not. When <i>GENESIS</i> encounters constraints with zero gradients it issues a warning message.

Information Switches

PARAMETER	POSSIBLE VALUES	DEFAULT	DEFINITION
IPRINT	Integer [0-7]	0	Print control for optimizer output during approximate optimization. Larger value gives more print. IPRINT=0 gives no print.

File Printing Controls

PARAMETER	POSSIBLE VALUES	DEFAULT	DEFINITION
GRAPH	0 1	0	Sets the format for the GRAPH solution control command. A value of 0 will set the format to HTML and the graph file will have the “.html” extension. A value of 1 will set the format to be postscript and the graph file will have the “.ps” extension.
OPTHIS	0 1	0	A value of 1 will cause the program to print the “*.OPT” file. A value of 0 will cause the program to not print the “*.OPT” file.
OPTGRID	0 1	1	Switch to determine how GRID data is written in the *.OPT file. A value of 1 will cause GRID data to be written using the same input coordinate system as in the data file. A value of 0 will cause GRID data to be written in the basic coordinate system.
OPOST	0 1	0	A value of 1 will cause the program to print all property fields in the OPOST post-processing file. A value of 0 will cause the program to only print a record for a given property field if at least one property has sizing data associated to that field.

Parameters for Discrete Variable Optimization

PARAMETER	POSSIBLE VALUES	DEFAULT	DEFINITION
DSCDOT	0 1 2	0	Parameter to control which optimizer is used for discrete variable optimization. If DSCDOT = 0, <i>GENESIS</i> will use BIGDOT. If DSCDOT = 1, <i>GENESIS</i> will use the DSCDOT optimizer. If DSCDOT = 2, <i>GENESIS</i> will use the CMBDOT optimizer.

DSTART	Integer ≥ -2	-1	<p>Parameter to control when discrete optimization starts.</p> <p>A value of -2 will make <i>GENESIS</i> ignore discrete variable sets (all DVARS will be continuous).</p> <p>A value of -1 will make <i>GENESIS</i> start the discrete optimization process after the continuous optimization process has converged.</p> <p>A value of 0 will make <i>GENESIS</i> start the discrete optimization process from the design cycle 0.</p> <p>A value of $N > 0$ will make <i>GENESIS</i> start the discrete optimization process after N continuous optimization cycles have been completed. If the continuous optimization process finishes in M design cycles with M less than N, then the discrete optimization process is started in the M+1 design cycle. Default = -1.</p>
DVINIT2	0 1 2 3	0	<p>Parameter used to control the starting design variable values.</p> <p>A value of 0, will cause the program to stop if the initial value of the design variable (INIT) does not belong to the associated discrete set.</p> <p>A value of 1 will cause the program to use the initial value of the design variable (INIT) even if it does not belong to the provided discrete set.</p> <p>A value of 2 will cause <i>GENESIS</i> to replace the design variable initial value with the value from the discrete set closest to the provided value.</p> <p>A value of 3 will cause <i>GENESIS</i> to replace the design variable initial value with the value from the discrete set closest to the provided value and then hold discrete design variables as constants during the optimization. This option allows restarts of a problem that has both discrete and continuous variables to keep improving the design using only the continuous variables.</p> <p>This parameter is ignored if DSTART=-2.</p>
IPEN	0 1	1	<p>Parameter used to control the discrete variable optimization method inside BIGDOT. 0 uses quadratic penalty. 1 uses sin function penalty.</p>
NDISCR	Integer > 0	1	<p>Parameter used to control the number of discrete optimization trials of BIGDOT.</p>
PENLTD	Real > 0.0	100000.0	<p>Initial penalty parameter used by BIGDOT during the discrete optimization process for discrete variable constraints. The parameter is updated by the PMULTD parameter using the following equation: $PENLTD(new) = PENLTD(old) * PMULTD$.</p>
PMULTD	Real > 0.0	5.0	<p>Penalty parameter multiplier for discrete variable constraints. See DOPT parameter PENLTD (above) for more details.</p>
MAXDPG	Integer > 0	5	<p>Maximum number of discrete variables in a segment. The larger MAXDPG is the larger the number of function call CMBDOT will make. Unless function evaluations are very inexpensive it is recommended to not exceed 10.</p>

Discrete Variables Move Limit Parameters (excluding composites)

PARAMETER	POSSIBLE VALUES	DEFAULT	DEFINITION
DDELP	Real > 0.0	0.5	Fractional change allowed for each designed property (except for composite properties) during the approximate optimization. This provides move limits.
DDPMIN	Real > 0.0	0.1	Minimum move limit factor imposed for properties (except for composite properties). The minimum move limit is defined as: Minimum Move Limit = $\begin{cases} \text{DDPMIN} & \text{if } P_{\text{initial}} \leq 1.0 \\ \text{DDPMIN} P_{\text{initial}} & \text{if } P_{\text{initial}} > 1.0 \end{cases}$

Discrete Variables Move Limit Parameters (For composites)

PARAMETER	POSSIBLE VALUES	DEFAULT	DEFINITION
DDELL	Real > 0.0	0.5	Fractional change allowed for thicknesses of composites designed property during the approximate optimization. This provides move limits.
DDELA	Real > 0.0	0.5	Fractional change allowed for angles of composites designed property during the approximate optimization. This provides move limits.
DDELC	Real > 0.0	0.5	Fractional change allowed for intermediate design variables of composites during the approximate optimization. This provides move limits.
DDLMIN	Real > 0.0	0.1	Minimum move limit factor imposed for thicknesses of composites properties. The minimum move limit is defined as: Minimum Move Limit = $\begin{cases} \text{DDLMIN} & \text{if } P_{\text{initial}} \leq 1.0 \\ \text{DDLMIN} P_{\text{initial}} & \text{if } P_{\text{initial}} > 1.0 \end{cases}$
DDAMIN	Real > 0.0	0.1	Minimum move limit factor imposed for angles of composites properties). The minimum move limit is defined as: Minimum Move Limit = $\begin{cases} \text{DDAMIN} & \text{if } P_{\text{initial}} \leq 1.0 \\ \text{DDAMIN} P_{\text{initial}} & \text{if } P_{\text{initial}} > 1.0 \end{cases}$
DDCMIN	Real > 0.0	0.1	Minimum move limit factor imposed for composite intermediate design variables properties. The minimum move limit is defined as: Minimum Move Limit = $\begin{cases} \text{DDCMIN} & \text{if } P_{\text{initial}} \leq 1.0 \\ \text{DDCMIN} P_{\text{initial}} & \text{if } P_{\text{initial}} > 1.0 \end{cases}$

Parameters for Stress Ratio

PARAMETER	POSSIBLE VALUES	DEFAULT	DEFINITION
ISRMET	0 1 2 3 4	0	The ISRMET parameter is used to control the stress ratio resizing method. If ISRMET=0, <i>GENESIS</i> will not use stress ratio. If ISRMET = 1 or 3, <i>GENESIS</i> will not change the design variables that reference elements that do not have stress ratio constraints. The difference between 1 and 3 is that in the first case <i>GENESIS</i> would stop due to hard convergence. Between 1 and 3, 3 is recommended for most cases. Option 1 should be picked over 3 only when analysis is very time consuming. If ISRMET=2 or 4, <i>GENESIS</i> will attempt to change the design variables of all designable rod and shell elements. The difference between 2 and 4 is that in the first case <i>GENESIS</i> would stop due to hard convergence. Between 2 and 4, 4 is recommended for most cases. Option 2 should be picked over 4 only when analysis is very time consuming.
ISRMAX	Integer > 0	3	The maximum number of stress ratio design cycles.

Parameters for STRDOT

PARAMETER	POSSIBLE VALUES	DEFAULT	DEFINITION
STRDOT	0 -1 1	0	The STRDOT parameter is used to control the use of the STRDOT optimizer. If STRDOT=0, <i>GENESIS</i> will not use the STRDOT optimizer. If STRDOT= -1 <i>GENESIS</i> will use the STRDOT optimizer and no other optimizer.. If STRDOT = 1, <i>GENESIS</i> will use the STRDOT optimizer first and then after finishing (after convergence or the number of design cycles exceeds ISDMAX) will switch to use the DOT or BIGDOT optimizer.
ISDMAX	Integer > 0	0	The maximum number of STRDOT design cycles.

Parameters for Reliability Based Optimization

PARAMETER	POSSIBLE VALUES	DEFAULT	DEFINITION
RBOOST	Integer ≥ -1	-1	Parameter to control when reliability optimization starts. A value of -1 will make <i>GENESIS</i> start the reliability optimization process after the deterministic optimization process has converged. A value of N will make <i>GENESIS</i> start the reliability optimization process after N deterministic optimization cycles have been completed. If the deterministic optimization process finishes in M design cycles with M less than N, then the probabilistic optimization process is started in the M+1 design cycle. Default = -1.

Parameters for Symmetry Constraints in Topometry Optimization

PARAMETER	POSSIBLE VALUES	DEFAULT	DEFINITION
SYMTOL	Real	0.001	Default tolerance for checking fabrication symmetry constraints in DSPLIT.
SYMMET	1 2	2	Default symmetry validation method for DSPLIT. 1: Center location 2: Corner grids of elements and center locations

Parameters for Matching Responses (DMATCH and DMATCH2)

PARAMETER	POSSIBLE VALUES	DEFAULT	DEFINITION
IMATCH	0 1	0	If IMATCH = 0, then the least square method is used with If IMATCH = 1, the Beta method is used. This parameter is used for DMATCH and DMATCH2
RMATCH	Real > 0.0	0.01	Minimum response/target normalizing parameter for DMATCH and DMATCH2. See Matching Analysis Results (p. 23)

Parameters for Nonlinear Contact Analysis Optimization

PARAMETER	POSSIBLE VALUES	DEFAULT	DEFINITION
CNTDC	Integer > 0	1	<p>Defines in which design cycles the program will perform full contact analysis. A value of n means that full contact analysis is performed every n-th design cycle. Regardless of the value of CNTDC, the program will always perform full contact analysis in the first and last design cycles.</p> <p>A value of 1, the default, causes the program to perform full contact analysis in every design cycle.</p> <p>A value greater than 1 will allow the program to use unconverged contact analyses in the skipped design cycles. The maximum number of contact iterations that the program will perform for the skipped design cycles is defined by the parameter CNTIT. For more information, See Nonlinear Contact Analysis Optimization (p. 28)</p>
CNTIT	Integer > 0	1	<p>Defines a maximum number of contact iterations that the program will perform in the design cycles that are not performing full converged contact analyses. This parameter is only used when CNTDC > 1.</p>

Other Parameters

PARAMETER	POSSIBLE VALUES	DEFAULT	DEFINITION
DESMIN	Integer > -1	0	Minimum number of design cycles. A value n, greater than zero, will force the program to perform at least n design cycles before stopping.
ITMAX	Integer > 0	100	Maximum number of iterations in the approximate optimization.
ITRMOP	Integer > 0	2	<p>The number of consecutive iterations DOT must satisfy the absolute or relative convergence criteria before optimization is terminated. Usually ITRMOP should be at least 2 because it is common to make little progress on one iteration, only to make major progress on the next. The default ITRMOP = 2 will allow a second try before terminating. If progress toward the optimum seems slow, but consistent, and function evaluations are not too expensive, it may improve the solution to increase ITRMOP to a value of 3 to 10.</p>

DVINIT	-1 0 1	0	A value of -1 will cause <i>GENESIS</i> to reset the initial value of all design variables to their lower bound. A value of 1 will cause <i>GENESIS</i> to set all design variables to their upper bounds. A value of 0 will leave the design variables at the original initial value.
BDMEM	$0.0 \leq \text{Real} \leq 100.0$	0.0	Percentage of free memory in the approximate module that will be additionally reserved for BIGDOT.
RPERT1	$0.0 \leq \text{Real}$	0.0	Controls the relative value of the perturbation cut-off. Perturbations (used in shape optimization) with a relative magnitude (compared to the largest perturbation using the same design variable) less than RPERT1 will be ignored.
RPERT2	$0.0 \leq \text{Real}$	0.0	Controls the absolute value of the perturbation cut-off. Perturbations (used in shape optimization) with an absolute magnitude less than RPERT2 will be ignored.
FREQREG	Integer $\neq 0$	1	Define how responses from different loading frequencies are grouped into screening regions. If FREQREG > 0 then FREQREG corresponds to the maximum number of dynamic loading frequencies per region. If FREQREG < 0 then ABS(FREQREG) corresponds to the maximum number of regions per frequency response loadcase. The most conservative screening method corresponds to FREQREG=1 and the most aggressive screening method corresponds to FREQREG=-1. An aggressive choice can cause the program to retain fewer responses, which will reduce sensitivity calculation time. However an aggressive choice may cause the program to need more design cycles to converge.
SENTHR	Integer > 0	*	Maximum number of parallel threads to use in sensitivity analysis. The default is to use the number defined by the THREADS executive control command.

RESETMV	Integer $\neq 0$	*	<p>Defines when or how often move limits are restored to their original values.</p> <p>If RESETMV > 0 then RESETMV corresponds to the design cycle number on which move limits are reset. This is a one time event. If the program converged before RESETMV number of design cycles, then this parameter does not take effect.</p> <p>If RESETMV < 0 then ABS(RESETMV) corresponds to the frequency on which move limits are reset. If N is the frequency of resetting, then the program will reset the move limits every N design cycle.</p> <p>The default will cause the program to not reset move limits in a given run.</p> <p>Note: move limits can be changed by automatic move limits adjustments. It should be noted that restarting the program also resets the move limits starting from the restarted design cycle.</p>
---------	------------------	---	---

8.2.12 DRESP1

Data Entry: **DRESP1** - Select Fundamental Response Quantities.

Description: Define a set of structural responses that is used in the design either as constraint or as an objective.

Format:

1	2	3	4	5	6	7	8	9	10
DRESP1	ID	LABEL	RTYPE	PTYPE	REGION	ATTa	ATT1	ATT2	ATT3
+	ATT4	-etc.-							

Alternate Format 1:

1	2	3	4	5	6	7	8	9	10
DRESP1	ID	LABEL	GRTYPE		REGION	ATTa	"GSET"	GSID	

Alternate Format 2:

1	2	3	4	5	6	7	8	9	10
DRESP1	ID	LABEL	ERTYPE	"ELEM"	REGION	ATTa	"ESET"	ESID	

Examples:

1	2	3	4	5	6	7	8	9	10
DRESP1	701	STR1	STRESS	PROD	2	1	15	102	103
+	110								

1	2	3	4	5	6	7	8	9	10
DRESP1	702	Z_DISP	DISP			3	GSET	55	

1	2	3	4	5	6	7	8	9	10
DRESP1	703	STR2	STRESS	ELEM		2	ESET	77	

Field Information Description

2	ID	Unique identification number. (Integer > 0). See remark 1.
3	LABEL	User defined label (or blank).

4	RTYPE	Response type. DISP, RELDISP, SPCF, STRESS, GSTRESS, STRAIN, FINDEX, CSTRESS, CSTRAIN, CTHICK, LTHICK, FORCE, CDISP, CPRESS, FREQ, RFREQ, ERECT, REVECT, LAMA, MASS, MASSFR, VOLUME, SENERGY, VMINDEX, TEMP, HTC, DDISP, DVELO, DACCE, DDISPS, DVELO, DACCES, MDISP, MVELO, MACCE, MDISPS, MVELO, MACCES, DSTRESS, DSTRAIN, DFORCE, DSTRS, DSTNS, DFORCES, PSDDISP, PSDVELO, PSDACCE, PSDDS, PSDVS, PSDAS, PSDSTR, RMSDISP, RMSVELO, RMSACCE, RMSSTR, DGSTRESS, UFDISP, UFVELO, UFACCE, UFDISPS, UFVELO, UFACCES, ERP or ERPS. See table below.
5	PTYPE	If RTYPE is STRAIN, DSTRAIN, DSTNS, STRESS, DSTRESS, DSTRS, PSDSTR, RMSSTR, FINDEX, CSTRESS, CSTRAIN, CTHICK, LTHICK, FORCE, DFORCE, DFORCES, MASS, MASSFR or VOLUME, this field is used to specify whether the ATTi are PID's, EID's or MID's. If PTYPE = ELEM, then the ATTi are EID's. If PTYPE is blank or PROP, PROD, PBEAM, PBAR, PSHELL, PCOMP, PSOLID, PSHEAR, PELAS, PDAMP, PVECTOR, PBUSH, or PVISC, then the ATTi are PID's (or Blank). If PTYPE = MAT, then the ATTi are the MID's. This field should be blank if RTYPE is DISP, RELDISP, SPCF, GSTRESS, CDISP, CPRESS, TEMP, VMINDEX, DGSTRESS, DDISP, DVELO, DACCE, DDISPS, DVELO, DACCES, MDISP, MVELO, MACCE, MDISPS, MVELO, MACCES, PSDDISP, PSDVELO, PSDACCE, PSDDS, PSDVS, PSDAS, RMSDISP, RMSVELO, RMSACCE, UFDISP, UFVELO, UFACCE, UFDISPS, UFVELO, UFACCES, ERP, ERPS. or INERTIA. If RTYPE = FREQ, RFREQ, ERECT or REVECT, then PTYPE = Integer corresponding to the mode number. If RTYPE = LAMA, then PTYPE = Integer corresponding to the buckling mode number. If RTYPE = SENERGY, then PTYPE = Integer corresponding to property IDs. See the table below.
6	REGION	Region identifier for constraint screening. (Integer > 0 or blank) (See Remarks 6 and 7 for defaults).
7	ATTA	Displacement component number(s), item number or eigenvector component. See table below and Remarks 13-18. (Integer > 0 or blank; blank if RTYPE is MASS, MASSFR, VOLUME, VMINDEX, FREQ, RFREQ, LAMA, TEMP or HTC).
8	ATT1	Grid, Spoint, Field Point, Property, Element or Material numbers, (Integer > 0 or blank; blank only if RTYPE is FREQ, RFREQ, VMINDEX, INERTIA or HTC). See table below. The keyword "ALL" can be used with grid responses: displacement, grid point stresses, temperatures or eigenvector components. See Remark 10. For shifted responses (RTYPE=DDISPS, DVELO, DACCES, MDISPS, MVELO, MACCES, PSDDS, PSDVS, PSDAS, UFDISPS, UFVELO, UFACCES, ERPS, DSTRS, DSTNS or DFORCES) use a DSHIFT identification number. For composite layer results (RTYPE=CSTRESS, CSTRAIN or LTHICK) use a layer number.

9	ATT2	Grid, Spoint, Field Point, Property, Element or Material numbers, (Integer > 0 or blank). See table below. The keyword "THRU" can be used with static displacement, static grid point stresses, temperatures or eigenvector components. See Remark 11.
10,2,3,...	ATTi	Grid, Spoint, Field Point, Property, Element, Material or Element Set IDs, (Integer > 0 or blank). See table below.
4	GRTYPE	Grid response type. DISP, RELDISP, SPCF, GSTRESS, CDISP, CPRESS, ETECT, REVECT, TEMP, DDISP, DVELO, DACCE, MDISP, MVELO, MACCE, PSDDISP, PSDVELO, PSDACCE, RMSDISP, RMSVELO, RMSACCE or DGSTRESS.
9	GSID	Set ID of a grid set defined with the SET solution control command.
4	ERTYPE	Element response type. STRESS, STRAIN, FINDEX, FORCE, DSTRESS, DSTRAIN, DFORCE, PSDSTR or RMSSTR.
9	ESID	Set ID of an element set defined with the SET solution control command.

RESPONSE	RTYPE	PTYPE	ATTA	ATTi
Static Displacement	DISP	Blank	Displacement component code. See remark 5.	Grid or Spoint ID or "GSET", "THRU" or "ALL"
Relative Static Displacement	RELDISP	Blank	Displacement component code.	Grid or Spoint ID
Reaction Force	SPCF	Blank	Displacement component code.	Grid or Spoint ID
Static Element Stress	STRESS	Blank or PROP, PAXIS, PBAR, PBEAM, PBUSH, PROD, PSHELL, PSOLID, PELAS, PSHEAR, ELEM	Stress item code	Property entry (PID) or Element ID (EID) or "ESET"
Dynamic Element Stress	DSTRESS	Blank or PROP, PAXIS, PBAR, PBEAM, PBUSH, PROD, PSHELL, PSOLID, PELAS, PSHEAR, ELEM	Stress item code	Property entry (PID) or Element ID (EID) or "ESET"

RESPONSE	RTYPE	PTYPE	ATTA	ATTi
Shifted Dynamic Element Stress	DSTRS	Blank or PROP, PAXIS, PBAR, PBEAM, PBUSH, PROD, PSHELL, PSOLID, PELAS, PSHEAR, ELEM	Stress Magnitude item code	ATT1: DSHIFT ID. ATTi (i>1): Property entry (PID) or Element ID (EID)
Random Power Spectral Density Dynamic (PSD) Element Stresses (From Direct or Modal Loadcases)	PSDSTR	Blank or PROP, PSHELL, PSOLID, ELEM	Component code.	Property entry (PID) or Element ID (EID) or "ESET"
Random RMS Element Stresses (From Direct or Modal Loadcases)	RMSSTR	Blank or PROP, PSHELL, PSOLID, ELEM	Component code.	Property entry (PID) or Element ID (EID) or "ESET"
Static Grid Stress	GSTRESS	Blank	Stress item code	Grid ID or "GSET", "THRU" or "ALL"
Dynamic Grid Stress	DGSTRESS	Blank	Stress item code	Grid ID or "GSET", "THRU" or "ALL"
Composite Layer Stress	CSTRESS	Blank, PCOMP or ELEM	Stress item code	ATT1: layer number. ATTi (i>1): Property entry (PID) or Element ID (EID)
Composite Element Failure Index	FINDEX	Blank, PCOMP or ELEM	Findex item code	Property entry (PID) or Element ID (EID) or "ESET"

RESPONSE	RTYPE	PTYPE	ATTA	ATTi
Static Element Strain	STRAIN	Blank or PROP, PAXIS, PBUSH, PSHELL, PSOLID, ELEM	Strain item code	Property entry (PID) or Element ID (EID) or "ESET"
Dynamic Element Strain	DSTRAIN	Blank or PROP, PAXIS, PBUSH, PSHELL, PSOLID, ELEM	Strain item code	Property entry (PID) or Element ID (EID) or "ESET"
Shifted Dynamic Element Strain	DSTNS	Blank or PROP, PAXIS, PBUSH, PSHELL, PSOLID, ELEM	Strain Magnitude item code	ATT1: DSHIFT ID. ATTi (i>1): Property entry (PID) or Element ID (EID)
Composite Layer Strain	CSTRAIN	Blank, PCOMP or ELEM	Strain item code	ATT1: layer number. ATTi (i>1): Property entry (PID) or Element ID (EID)
Static Element Force	FORCE	Blank or PROP, PBAR, PBEAM, PBUSH, PSHELL, PELAS, PSHEAR, PROD, PVECTOR, ELEM	Force item code	Property entry (PID) or Element ID (EID) or "ESET"
Dynamic Element Force	DFORCE	Blank or PROP, PBAR, PBEAM, PBUSH, PCOMP, PSHELL, PELAS, PSHEAR, PROD, PDAMP, PVECTOR, PVISC, ELEM	Force item code	Property entry (PID) or Element ID (EID) or "ESET"
Shifted Dynamic Element Force	DFORCES	Blank or PROP, PBAR, PBEAM, PBUSH, PCOMP, PSHELL, PELAS, PSHEAR, PROD, PDAMP, PVECTOR, PVISC, ELEM	Force Magnitude item code	ATT1: DSHIFT ID. ATTi (i>1): Property entry (PID) or Element ID (EID)

RESPONSE	RTYPE	PTYPE	ATTA	ATTi
Composite Total Thickness	CTHICK	Blank, PCOMP or ELEM	1	Property entry (PID) or Element ID (EID)
Composite Layer Thickness	LTHICK	Blank, PCOMP or ELEM	Thickness item code: 1 = layer thickness 2 = layer thickness / total thickness	ATT1: layer number. ATTi (i>1): Property entry (PID) or Element ID (EID)
Grid Contact Clearance	CDISP	Blank	Contact component code: 1	Grid ID or "GSET", "THRU" or "ALL"
Grid Contact Pressure or Grid Glue Connection Pressure	CPRESS	Blank	Contact component code: 1	Grid ID or "GSET", "THRU" or "ALL"
Natural Frequency	FREQ, RFREQ	Mode number	Blank	Blank
Eigenvector Component	EVECT, REVECT	Mode number	Component code	Grid or Spoint ID or "GSET", "THRU" or "ALL"
Buckling Load Factor	LAMA	Mode number	Blank	Blank
Mass	MASS	Blank or MAT or PROP	Blank	Blank or Property entry (PID) or Material entry (MID)
Mass fraction	MASSFR	Blank or MAT or PROP	Blank	Blank or Property entry (PID) or Material entry (MID)

RESPONSE	RTYPE	PTYPE	ATTA	ATTi
Volume	VOLUME	Blank or MAT or PROP	Blank	Blank or Property entry (PID) or Material entry (MID)
Static Strain Energy	SENERGY	Blank or PROP or PAXIS or PBAR or PBARL or PBEAM or PBEAML or PBUSH or PROD or PSHELL or PSHEAR or PCOMP or PCOMPG or PSOLID or PELAS or PVECTOR or PBUSH or PBUSHT or PGARP or PWELD or PK2UU	Blank	Blank
Von Mises Stress Index	VMINDEX	Blank	Blank	Blank
Temperature	TEMP	Blank	Blank	GRID or SPOINT ID or "GSET", "THRU" or "ALL"
Heat Transfer Compliance	HTC	Blank	Blank	Blank
Dynamic Displacement, Velocity, Acceleration (From Direct Loadcases)	DDISP, DVELO, DACCE	Blank	Component code (1-24)	Grid or Spoint ID or "GSET"
Shifted Dynamic Displacement, Velocity, Acceleration (From Direct Loadcases)	DDISPS, DVELO, DACCES	Blank	Component code;(1-6)	ATT1: DSHIFT ID. ATTi (i>1): Grid or Spoint ID
Dynamic Displacement, Velocity, Acceleration (From Modal Loadcases)	MDISP, MVELO, MACCE	Blank	Component code (1-24)	Grid or Spoint ID or "GSET"

RESPONSE	RTYPE	PTYPE	ATTA	ATTi
Shifted Dynamic Displacement, Velocity, Acceleration (From Modal Loadcases)	MDISPS, MVELO, MACCES	Blank	Component code: (1-6)	ATT1: DSHIFT ID. ATTi (i>1): Grid or Spoint ID
Random Power Spectral Density Dynamic Displacement, Velocity, Acceleration (From Direct or Modal Loadcases)	PSDDISP, PSDVELO, PSDACCE	Blank	Component code.	Grid or Spoint ID
Shifted Random Power Spectral Density Dynamic Displacement, Velocity, Acceleration (From Direct or Modal Loadcases)	PSDDDS, PSDVLS, PSDAS	Blank	Component code: (1-6)	ATT1: DSHIFT ID. ATTi (i>1): Grid or Spoint ID
Random Dynamic Displacement, Velocity, Acceleration (From Random Loadcases)	RMSDISP, RMSVELO, RMSACCE	Blank	Component code (1-6)	Grid or Spoint ID
User Function of Dynamic Displacement, Velocity, Acceleration	UFDISP, UFVELO, UFACCE	Blank	Field Point item code 1: Magnitude 2: Phase 3: Real 4: Imaginary	Field Point ID (Defined in UFDATA file)

RESPONSE	RTYPE	PTYPE	ATTA	ATTi
Shifted User Function of Dynamic Displacement, Velocity, Acceleration (From Direct or Modal Loadcases)	UFDISPS, UFVELOS, UFACCES	Blank	Component code 1: Magnitude	ATT1: DSHIFT ID. ATTi (i>1):Field Point ID (Defined in UFDATA file)
Equivalent Radiated Power	ERP	Blank	Item code 1: Actual ERP value 2: Decibel scale value	Element set ID (Specified in ERPPNL entry)
Shifted Equivalent Radiated Power	ERPS	Blank	Item code 1: Actual ERP value 2: Decibel scale value	ATT1: DSHIFT ID. ATTi (i>1): Element set ID (Specified in ERPPNL entry)
System Inertia Matrix Component	INERTIA	Blank	Inertia item code	Blank

Remarks:

1. **DRESP1** identification numbers must be unique with respect to **DRESP2**, **DRESP3**, **DRESPG** and **DRESPU** identification numbers.
2. All the elements referenced by a DRESP1 statement, either directly or indirectly through their property data, must be of the same element class (i.e. ELAS1, ROD, BAR, SHELL, SOLID, VISC or DAMP1). Element types of the same class can be mixed, i.e. CTRIA3, CTRIA6, QUAD4 and CQUAD8 or CHEXA, CPENTA, CPYRA and CTETRA. Elements whose properties are controlled by DVPROP3 design variable to property relations cannot be mixed with elements that are not controlled by DVPROP3 entries.
3. If PTYPE = ELEM the ATTi correspond to element identification numbers.
4. If PTYPE = Blank or PROP, PROD, PBEAM, PBAR, PBUSH, PSHELL, PCOMP, PSHEAR, PSOLID, PELAS, PVECTOR, PDAMP or PVISC, the responses will be generated for **all** the elements that reference each of the PIDs listed in the ATTi.
5. For displacement and grid point stress responses, all grids associated with a DRESP1 entry are considered to be in the same region for screening purposes. Only a limited number (NSTR from the DSCREEN data) of displacement, temperature or grid point stress constraints per region per load case (per loading frequency) will be retained in the design optimization phase.

6. REGION is used for constraint screening. The NSTR field on DSCREEN entries gives the maximum number of constraints retained for each region per load case (per loading frequency). If RTYPE = MASS, MASSFR, VOLUME, SENERGY, FREQ, RFREQ or LAMA no REGION identification number should be specified. For all other responses, if the REGION field is left blank, the default specified in the table below is used. Usually, the default value is appropriate. If the REGION field is not blank, all the responses on this entry, as well as all responses on other DRESP1 entries which have the same RTYPE, element type, and REGION identification number will be grouped into the same region.

RESPONSE TYPE	DEFAULT REGION
MASS	No region
MASSFR	No region
VOLUME	No region
VMINDEX	No region
INERTIA	No region
FREQ, RFREQ	No region
LAMA	No region
SENERGY	No region
HTC	No region

DRESP1

Shape, Sizing, Topography, Topometry and Freeform Design Model Data

RESPONSE TYPE	DEFAULT REGION
DISP, RELDISP, SPCF, DDISP, DDISPS, MDISP, MDISPS, PSDDISP, PSDDS RMSDISP, DVELO, DVELOS, MVELO, MVELOS, PSDVELO, PSDVS, RMSVELO, DACCE, DACCES, MACCE, MACCES, PSDACCE, PSDAS, RMSACCE UFDISP, UFDISPS, UFVELO, UFVELOS, UFACCE UFACCES ERP ERPS	One region per DRESP1 Entry
CDISP CPRESS	One region per DRESP1 Entry
EVECT, REVECT	One region per DRESP1 entry

RESPONSE TYPE	DEFAULT REGION
EVECT, REVECT	One region per DRESP1 entry
TEMP	One region per DRESP1 Entry
ALL OTHERS	One region per property entry. If PTYPE = ELEM, then one region per DRESP1 Entry

7. REGION is valid only among the same type of responses. Responses of different types will never be grouped into the same region, even if they are assigned the same REGION identification number by the user. Responses of different types will be grouped in different regions, even if the same REGION ID is given.
8. If RTYPE = RFREQ or REVECT, responses can only be generated for reduced eigenvalue loadcases (ASET Loadcases). If these responses are selected, then *GENESIS* will calculate the frequency and eigenvector sensitivities using the reduced problem. In other words, *GENESIS* will calculate the sensitivities exactly.
9. If RTYPE = FREQ or EVECT, responses can be generated for all eigenvalue loadcases. If these responses are used on a reduced loadcase, then *GENESIS* will calculate the frequency and eigenvector sensitivities using the full size problem. In other words, the sensitivities of the reduced problem will be calculated approximately. For normal (non- ASET) eigenvalue loadcases, frequency and eigenvector sensitivities are always calculated exactly.
10. The keyword “ALL” can be used with displacement, grid point stresses, grid contact pressure, temperatures and eigenvectors to specify all of the grid points. For example:

1	2	3	4	5	6	7	8	9	10
DRESP1	10	HOT	TEMP				ALL		

11. The keyword “THRU” can be used with displacement, grid point stresses, grid contact pressure, temperatures and eigenvectors to specify the grid points. For example:

1	2	3	4	5	6	7	8	9	10
DRESP1	10	FACE2	GSTRESS			2	1	THRU	100
+	200	300	THRU	320					

specifies grid points 1, 2, 3, . . . , 100, 200, 300, 301, 302, . . . , 320.

12. Scalar points cannot be listed for grid point stress responses or grid contact pressure. Scalar points can be listed for temperature responses. Scalar points are always ignored by the “ALL” and “THRU” commands.
13. If RTYPE=DISP, EVECT or REVECT, multiple components (1-6) can be specified on a single entry. Multiple response components cannot be used on any other response types. The component should be 1 for scalar points. For RTYPE = DISP, component 7 is defined to be a spherical displacement constraint and cannot be

combined with other components. The displacements are in the output coordinate system; in other words, they are in the local coordinate system defined in field 7 of GRID data. If that field is blank the displacements are output in the basic coordinate system.

14. If RTYPE = DDISP, DVELO, DACCE, MDISP, MVELO or MACCE, then only one component can be specified. The components for grid points are 1-6 for magnitude, 7-12 for phase, 13-18 for real and 19-24 for imaginary. The component code for scalar points for dynamic responses should be 1 for magnitude, 7 for phase, 13 for real component and 19 for imaginary component.
15. If RTYPE = DDISPS, DVELOs, DACCES, MDISPS, MVELOs or MACCES, then only one component can be specified. The components for grid points are 1-6. The component code for scalar points for dynamic responses should be 1.
16. If RTYPE = PSDDISP, PSDVELO, PSDACCE, PSDDS, PSDVS or PSDAS, then only one component can be specified. The components for grid points are 1-6. The component code for scalar points for dynamic responses should be 1.
17. If RTYPE = FINDEX, the response is either the composite ply failure index (ATTA=1) using the failure theory specified on the PCOMP data or the composite interlaminar bonding shear failure index (ATTA=2).
18. Stress, Strain, Force and Grid Stress item codes are listed below. Note that for dynamic analysis, you may use any of magnitude, phase, real or imaginary columns, while for static analysis, only the magnitude column is used. Also, note that in dynamic analysis, the magnitude of a response is always positive but in static analysis the magnitude of a response can be either positive or negative.
19. If RTYPE = PSDSTR or RMSSTR, then only one component can be specified. The components for solid element stresses are 1-7 and for shell elements are: 2,5,6,7,9,12,13 & 14. See stress item code tables below for more details.
20. If RTYPE = UFDISP, UFVELO or UFACCE, the user function is defined by file specified by the UFDATA executive control command. The field point item codes are: 1 for magnitude, 2 for phase, 3 for real and 4 for imaginary.
21. If RTYPE = UFDISPS, UFVELOs, or UFACCES, the user function is defined by file specified by the UFDATA executive control command. The field point item code should be 1.
22. For shifted responses (RTYPE=DDISPS, DVELOs, DACCES, MDISPS, MVELOs, MACCES, PSDDS, PSDVS, PSDAS, UFDISP, UFVELOs, UFACCES, ERPS, DSTRS, DSTNS or DFORCES) the **DSHIFT** data entry is required.
23. If RTYPE = CPRESS, the response is the grid contact (or glue connection) pressure measured at the grid in the normal direction of the contact surface (ATTA=1). The contact pressure is calculated by dividing the normal contact force at the grid by the associated area.
24. For composite layer responses (RTYPE=CSTRESS, CSTRAIN or LTHICK), the layer number is counted up starting from 1 at the bottom layer.

25. If RTYPE = ERP or ERPS, the element set id must also be specified on an ERPPNL entry. The item code should be either 1 (for linear scale) or 2 (for decibel scale). For RTYPE=ERPS, if item code is 2, then the corresponding DSHIFT data can only use the FTYPE=IDENT or FTYPE=RECIP option (LOG or LOG10 option in DSHIFT can not be applied to ERP in decibel scale).
26. If RTYPE = HTC, the response is the heat transfer compliance, which is defined as $HTC = \frac{1}{2} \{T\}^T \{F\}$, where $\{T\}$ is the vector of grid temperatures, and $\{F\}$ is the vector of applied heat fluxes.
27. If RTYPE = VMINDEX, the response is the von Mises stress index, which is a global value that estimates the maximum von Mises stress in elements. The von Mises stress limit must be defined in the SS field of MAT1 entries. Elements that reference MAT1 entries that do not define SS are not included in the index. If the response value is greater than 1.0, then either some elements have a von Mises stress significantly higher than SS, or there are a significant number of elements with von Mises stress somewhat higher than SS. Note that because the index is merely an estimate, having the response value less than 1.0 does not guarantee that all elements have von Mises stress less than SS, but the number of violating elements should be a small fraction of the total number of elements. The VMINDEX estimate is most accurate for solid elements, and significantly less accurate for 1-D and 2-D elements.

FORCE ITEM CODES

ELAS1, ELAS2

Item #				Force
Static or Magnitude	Phase	Real Comp.	Imaginary Comp.	
1	2	3	4	Element force

BUSH

User Item #				Force
Static or Magnitude	Phase	Real Comp.	Imaginary Comp.	
1	7	13	19	Force x
2	8	14	20	Force y
3	9	15	21	Force y
4	10	16	22	Moment x
5	11	17	23	Moment y
6	12	18	24	Moment y

ROD

Item #				Force
Static or Magnitude	Phase	Real Comp.	Imaginary Comp.	
1	3	5	7	Axial force at end A
2	4	6	8	Axial force at end B

FORCE ITEM CODES**BAR**

Item #				Force
Static or Magnitude	Phase	Real Comp.	Imaginary Comp.	
1	13	25	37	Axial force at end A
2	14	26	38	Shear in plane 1 at end A
3	15	27	39	Shear in plane 2 at end A
4	16	28	40	Torque at end A
5	17	29	41	Moment in plane 2 at end A
6	18	30	42	Moment in plane 1 at end A
7	19	31	43	Axial force at end B
8	20	32	44	Shear in plane 1 at end B
9	21	33	45	Shear in plane 2 at end B
10	22	34	46	Torque at end B
11	23	35	47	Moment in plane 2 at end B
12	24	36	48	Moment in plane 1 at end B

DAMP1, DAMP2

Item #				Force
Static or Magnitude	Phase	Real Comp.	Imaginary Comp.	
1	2	3	4	Element force

VISC

Item #				Force
Static or Magnitude	Phase	Real Comp.	Imaginary Comp.	
1	3	5	7	Axial force in element
2	4	6	8	Torque in element

FORCE ITEM CODES

QUAD4, QUAD8, TRIA3 and TRIA6 (PSHELL Only)

Item #				Force
Static or Magnitude	Phase	Real Comp.	Imaginary Comp.	
1	9	17	25	N_x (In SCSID coordinate system)
2	10	18	26	N_y (In SCSID coordinate system)
3	11	19	27	N_{xy} (In SCSID coordinate system)
6	14	22	30	M_x (In SCSID coordinate system)
7	15	23	31	M_y (In SCSID coordinate system)
8	16	24	32	M_{xy} (In SCSID coordinate system)

QUAD4, QUAD8, TRIA3 and TRIA6 (PCOMP/PCOMPG Only)

Item #				
Static				
1				N_x (In material coordinate system)
2				N_y (In material coordinate system)
3				N_{xy} (In material coordinate system)
6				M_x (In material coordinate system)
7				M_y (In material coordinate system)
8				M_{xy} (In material coordinate system)

FORCE ITEM CODES**SHEAR**

Item #				Force
Static or Magnitude	Phase	Real Comp.	Imaginary Comp.	
1	17	33	49	Force 4 to 1
2	18	34	50	Force 2 to 1
3	19	35	51	Force 1 to 2
4	20	36	52	Force 3 to 2
5	21	37	53	Force 2 to 3
6	22	38	54	Force 4 to 3
7	23	39	55	Force 3 to 4
8	24	40	56	Force 1 to 4
9	25	41	57	Kick Force on 1
10	26	42	58	Shear 12
11	27	43	59	Kick Force on 2
12	28	44	60	Shear 23
13	29	45	61	Kick Force on 3
14	30	46	62	Shear 34
15	31	47	63	Kick Force on 4
16	32	48	64	Shear 41

STRESS ITEM CODES**ELAS1, ELAS2**

Item #				Stress
Static or Magnitude	Phase	Real Comp.	Imaginary Comp.	
1	2	3	4	Element stress (Force x SRC)

VECTOR SPRING

Item #				Response
Static or Magnitude	Phase	Real Comp.	Imaginary Comp.	
1	13	25	37	Force 1 at grid A
2	14	26	38	Force 2 at grid A
3	15	27	39	Force 3 at grid A
4	16	28	40	Moment 1 at grid A
5	17	29	41	Moment 2 at grid A
6	18	30	42	Moment 3 at grid A
7	19	31	43	Force 1 at grid B
8	20	32	44	Force 2 at grid B
9	21	33	45	Force 3 at grid B
10	22	34	46	Moment 1 at grid B
11	23	35	47	Moment 2 at grid B
12	24	36	48	Moment 3 at grid B

STRESS ITEM CODES**BUSH**

User Item #				Stress
Static or Magnitude	Phase	Real Comp.	Imaginary Comp.	
1	7	13	19	Stress x = ST*Fx
2	8	14	20	Stress y= ST*Fy
3	9	15	21	Stress x = ST*Fz
4	10	16	22	$\tau_x = SR*M_x$
5	11	17	23	$\tau_y = SR*M_y$
6	12	18	24	$\tau_z = SR*M_z$

ROD

Item #				Stress
Static or Magnitude	Phase	Real Comp.	Imaginary Comp.	
1	3	5	7	Axial stress at end A
2	4	6	8	Axial stress at end B

SHEAR

User Item #				Stress
Static or Magnitude	Phase	Real Comp.	Imaginary Comp.	
1	7	13	19	Shear stress at grid point 1
2	8	14	20	Shear stress at grid point 2
3	9	15	21	Shear stress at grid point 3
4	10	16	22	Shear stress at grid point 4
5	11	17	23	Average Shear Stress
6	12	18	24	Maximum Shear Stress

STRESS ITEM CODES**BAR not designed with DVPROP3**

Item #				Stress
Static or Magnitude	Phase	Real Comp.	Imaginary Comp.	
1	9	17	25	Bending + axial at location C at end A
2	10	18	26	Bending + axial at location D at end A
3	11	19	27	Bending + axial at location E at end A
4	12	20	28	Bending + axial at location F at end A
5	13	21	29	Bending + axial at location C at end B
6	14	22	30	Bending + axial at location D at end B
7	15	23	31	Bending + axial at location E at end B
8	16	24	32	Bending + axial at location F at end B

BAR designed with DVPROP3 Types 1-11 (Built-in bar cross-sections)

See the design element library ([DVPROP3](#)). The stress calculation points are shown there. The item code numbers for stress magnitude on a positive face of end A correspond to the locations numbers. The item code numbers for stress magnitude at end B correspond to the sum of location number and the total number of locations.

The item codes for the phases are obtained by adding two times the total number of locations to the corresponding item for magnitude. The item codes for the real components are obtained by adding four times the total number of locations to the corresponding item for magnitude. The item codes for imaginary stresses are obtained by adding six times the total number of locations to the corresponding item for magnitude.

BAR designed with DVPROP3 Types 15-25 (User-subroutine bar cross-sections)

The item codes for user supplied design elements correspond to the locations in the STRESS vector defined by the user in the GN40ii Subroutines. For example, the item code number for the stress that the user calculates in GN4020 as $STRESS(4)=SA1-SA2+SA3$ is 4. See: [User-Supplied Stress Recovery Calculation Interface Function \(DLIB STRESS\)](#) (p. 292). If the user subroutine GN40ii calculates six stresses per element (3 in face A and 3 in face B), then the item code numbers for dynamic analysis are 1-6 for the magnitudes, 7-12 for the phases, 13-18 for the real components and 19-24 for the imaginary components.

STRESS ITEM CODES

QUAD4, QUAD8, TRIA3 and TRIA6 (PSHELL Only) not designed with DVPROP3

User Item #							Surface	Stress
Static	Dynamic				Random			
	Magnitude	Phase	Real Comp.	Imaginary Comp.	PSD	RMS		
1	-	-	-	-	-	-	1	Max shear
2	-	-	-	-	2	2	1	von Mises
3	-	-	-	-	-	-	1	Principal 1
4	-	-	-	-	-	-	1	Principal 2
5	5	19	33	47	5	5	1	Normal x
6	6	20	34	48	6	6	1	Normal y
7	7	21	35	49	7	7	1	Shear xy
8	-	-	-	-	-	-	2	Max shear
9	-	-	-	-	9	9	2	von Mises
10	-	-	-	-	-	-	2	Principal 1
11	-	-	-	-	-	-	2	Principal 2
12	12	26	40	54	12	12	2	Normal x
13	13	27	41	55	13	13	2	Normal y
14	14	28	42	56	14	14	2	Shear xy

QUAD4, QUAD8, TRIA3 and TRIA6 (PSHELL Only) designed using DVPROP3 Types 12 - 14 (Built-in plate sections: SOLID, SAND, and SAND2)

Stress item codes are the same as for non-DVPROP3 PSHELL elements.

QUAD4, QUAD8, TRIA3 and TRIA6 (PSHELL Only) designed using DVPROP3 Types 15 - 25 (User-subroutine plate sections)

The item codes of the user supplied design elements corresponds to the location in the STRESS vector defined by the user in Subroutines GN40ii. See [User-Supplied Stress Recovery Calculation Interface Function \(DLIB STRESS\)](#) (p. 292).

For dynamic analysis, the user can specify item codes from 1 to $4n$, where n is the number of stresses calculated by the user subroutine. Items 1 to n are the stress magnitudes, items $n+1$ to $2n$ are the phases, items $2n+1$ to $3n$ are the real components and items $3n+1$ to $4n$ are the imaginary components.

STRESS and STRAIN ITEM CODES**QUAD4, QUAD8, TRIA3 and TRIA6 (PCOMP/PCOMPG Only)**

User Item #				
Static				
1				Max shear
2				von Mises
3				Principal 1
4				Principal 2
5				Normal x (in Layer i Coordinate System)
6				Normal y (in Layer i Coordinate System)
7				Shear xy (in Layer i Coordinate System)

STRAIN ITEM CODES**BUSH**

User Item #				Strain
Static or Magnitude	Phase	Real Comp.	Imaginary Comp.	
1	7	13	19	Strain x = $ET^*(U_{x2}-U_{x1})$
2	8	14	20	Strain y = $ET^*(U_{y2}-U_{y1})$
3	9	15	21	Strain x = $ET^*(U_{z2}-U_{z1})$
4	10	16	22	$\gamma_x = ER^*(\theta_{x2}-\theta_{x1})$
5	11	17	23	$\gamma_y = ER^*(\theta_{y2}-\theta_{y1})$
6	12	18	24	$\gamma_z = ER^*(\theta_{z2}-\theta_{z1})$

STRAIN ITEM CODES**QUAD4, QUAD8, TRIA3 & TRIA6 (PSHELL Only)**

User Item #					Surface	Strain
Static	Dynamics					
	Magnitude	Phase	Real Comp.	Imaginary Comp.		
1	-	-	-	-	1	Max shear
2	-	-	-	-	1	von Mises
3	-	-	-	-	1	Principal 1
4	-	-	-	-	1	Principal 2
5	5	27	49	71	1	Normal x
6	6	28	50	72	1	Normal y
7	7	29	51	73	1	Shear xy
8	-	-	-	-	2	Max shear
9	-	-	-	-	2	von Mises
10	-	-	-	-	2	Principal 1
11	-	-	-	-	2	Principal 2
12	12	34	56	78	2	Normal x
13	13	35	57	79	2	Normal y
14	14	36	58	80	2	Shear xy
15	15	37	59	81	Midplane	ϵ_x
16	16	38	60	82	Midplane	ϵ_y
17	17	39	61	83	Midplane	γ_{xy}
20	20	42	64	86	Midplane	κ_x
21	21	43	65	87	Midplane	κ_y
22	22	44	66	88	Midplane	κ_{xy}

GSTRESS, DGSTRESS, STRESS, STRAIN and DSTRAIN ITEM CODES

for TRIAX6

Item #					Stress: GSTRESS DSTRESS, DGSTRES,	Strain: Dstrain
Static	Dynamics					
	Magnitude	Phase	Real Comp.	Imaginary Comp.		
1	1	14	27	40	σ_r	ϵ_r
2	2	15	28	41	σ_θ	ϵ_θ
3	3	16	29	42	σ_z	ϵ_z
6	6	19	32	45	τ_{rz}	γ_{rz}
7	-	-	-	-	von Mises stress	von Mises Strain
8	-	-	-	-	Octahedral stress	Octahedral strain
9	-	-	-	-	Max shear stress	Max shear strain
10	-	-	-	-	Mean pressure	$\frac{\text{Delta volume}}{\text{Volume}}$
11	-	-	-	-	Principal 1	Principal 1
12	-	-	-	-	Principal 2	Principal 2
13	-	-	-	-	Principal 3	Principal 3

GSTRESS, DGSTRESS, STRESS and STRAIN ITEM CODES**for HEXA, HEX20, PENTA, PYRA, TETRA**

Item #							Stress: GSTRESS, DSTRESS, DGSTRES, PSDSTR, RMSTRS	Strain: Dstrain
Static	Dynamics				Random			
	Magnitude	Phase	Real Comp.	Imaginary Comp.	PSD	RMS		
1	1	14	27	40	1	1	Normal x	Normal x
2	2	15	28	41	2	2	Normal y	Normal y
3	3	16	29	42	3	3	Normal z	Normal z
4	4	17	30	43	4	4	Shear xy	Shear xy
5	5	18	31	44	5	5	Shear yz	Shear yz
6	6	19	32	45	6	6	Shear zx	Shear zx
7	-	-	-	-	7	7	von Mises stress	von Mises Strain
8	-	-	-	-	-	-	Octahedral stress	Octahedral strain
9	-	-	-	-	-	-	Max shear stress	Max shear strain
10	-	-	-	-	-	-	Mean pressure	$\frac{\text{Delta volume}}{\text{Volume}}$
11	-	-	-	-	-	-	Principal 1	Principal 1
12	-	-	-	-	-	-	Principal 2	Principal 2
13	-	-	-	-	-	-	Principal 3	Principal 3

FINDEX ITEM CODES

QUAD4, QUAD8, TRIA3 and TRIA6 (PCOMP/PCOMPG Only)

Item #				
Static				
1				Ply failure (FP)
2				Interlaminar shear failure (FB)

LAYER THICKNESS ITEM CODES**QUAD4, QUAD8, TRIA3 and TRIA6 (PCOMP/PCOMPG Only)**

Item #				
Static				
1				Actual layer thickness
2				Fraction: Layer thickness / Total thickness

CONTACT COMPONENT CODE
(Nonlinear Contact Analysis Only)

Item #				
Static				
1				Normal direction: CDISP - Clearance between contact surfaces CPRESS - Pressure normal to the contact surface or normal to glue connected surfaces

SYSTEM INERTIA ITEM CODES

Item #	Meaning
1	Ixx at center of gravity
2	Iyy at center of gravity
3	Izz at center of gravity
4	Ixy at center of gravity
5	Iyz at center of gravity
6	Izx at center of gravity
7	Principal 1 at center of gravity
8	Principal 2 at center of gravity
9	Principal 3 at center of gravity
10	Ixx at grdpnt
11	Iyy at grdpnt
12	Izz at grdpnt
13	Ixy at grdpnt
14	Iyz at grdpnt
15	Izx at grdpnt
16	Principal 1 at grdpnt
17	Principal 2 at grdpnt
18	Principal 3 at grdpnt
20	Y center of gravity with respect to grdpnt
21	Z center of gravity with respect to grdpnt
22	X center of gravity with respect to grdpnt

8.2.13 DRESP2

Data Entry: **DRESP2** - Create Equation Response Quantities.

Description: Define equation responses that are used in the design, either as an objective function or as constraints.

Format:

1	2	3	4	5	6	7	8	9	10
DRESP2	ID	LABEL	EQID	REGION					
+	"DVAR"	NDV1	NDV2	NDV3	NDV4	NDV5	NDV6	NDV7	NDV8
+		NDV9	-etc.-						
+	"DTABLE"	NC1	NC2	NC3	NC4	NC5	NC6	NC7	NC8
+		NC9	-etc.-						
+	"DGRID"	NG1	NGC1	NG2	NGC2	NG3	NGC3	NG4	NGC4
+		NG5	NGC5	-etc.-					
+	"DRESP1"	NR1	NR2	NR3	NR4	NR5	NR6	NR7	NR8
+		NR9	-etc.-						

Alternate Format 1:

1	2	3	4	5	6	7	8	9	10
DRESP2	ID	LABEL	EQID						
+	"DVAR"	NDV1	NDV2	NDV3	NDV4	NDV5	NDV6	NDV7	NDV8
+		NDV9	-etc.-						
+	"DTABLE"	NC1	NC2	NC3	NC4	NC5	NC6	NC7	NC8
+		NC9	-etc.-						
+	"DGRID"	NG1	NGC1	NG2	NGC2	NG3	NGC3	NG4	NGC4
+		NG5	NGC5	-etc.-					
+	"DRESP1L"	NR1	LIDR1	NR2	LIDR2	NR3	LIDR3	NR4	LIDR4
+		NR5	LIDR5	-etc.-					

Alternate Format 2:

1	2	3	4	5	6	7	8	9	10
DRESP2	ID	LABEL	EQID						
+	"DVAR"	NDV1	NDV2	NDV3	NDV4	NDV5	NDV6	NDV7	NDV8
+		NDV9	-etc.-						
+	"DTABLE"	NC1	NC2	NC3	NC4	NC5	NC6	NC7	NC8
+		NC9	-etc.-						
+	"DGRID"	NG1	NGC1	NG2	NGC2	NG3	NGC3	NG4	NGC4
+		NG5	NGC5	-etc.-					
+	"DRESP2"	NS1	NS2	NS3	NS4	NS5	NS6	NS7	NS8
+		NS9	-etc.-						

Example:

1	2	3	4	5	6	7	8	9	10
DRESP2	1	EBUCK	3	4					
+	DVAR	87	25	3	8				
+	DTABLE	VAL1	YOUNGS	CVAL4					
+	DGRID	2	3	4	2				
+	DRESP1	12	5	2	102	202	302	402	502
+		602	702						

Field Information Description

2	ID	Unique identification number. (Integer > 0). See remark 6.
3	LABEL	User defined label for information only (Characters or blank).
4	EQID	DEQATN entry identification number. (Integer > 0).
5	REGION	Region identifier for constraint screening. (Integer > 0 or blank. Default puts this set of responses in a unique region.).
2	DVAR	Word indicating DVAR identification numbers.
3,4,..	NDVi	DVAR identification number. (Integer > 0 or Blank or "THRU").
2	DTABLE	Word indicating constant identification numbers in DTABLE.
3,4,..	NCj	Name of constant defined in DTABLE input. (Character). See Remark 17.
2	DGRID	Word indicating GRID identification and component numbers. See remark 10.
3,5,..	NGn	GRID identification number (Integer > 0 or Blank).
4,6,..	NGCn	GRID component number ($1 \leq \text{Integer} \leq 3$ or Blank).

DRESP2

Shape, Sizing, Topography, Topometry and Freeform Design Model Data

2	DRESP1	Word indicating DRESP1 or DRESPG identification numbers.
3,4,..	NRk	DRESP1 or DRESPG identification number. (Integer > 0 or Blank or "THRU"). See remarks 2, 4 and 5.
2	DRESP1L	Word indicating DRESP1 or DRESPG and Loadcase identification numbers.
3,5,..	NRk	DRESP1 or DRESPG identification number (Integer > 0 or Blank). See remark 12.
4,6,..	LIDRk	Loadcase identification number (Integer > 0).
2	DRESP2	Word indicating DRESP2 or DRESP3 identification numbers.
3,4,..	NSk	DRESP2 or DRESP3 identification number. (Integer > 0 or Blank or "THRU").

Remarks:

1. DRESP2 entries can only reference DVAR, DTABLE, DRESP1, DRESP2, DRESP3, DRESPG and GRID entries. They cannot reference DRESPU entries.
2. If the DRESP1 keyword is used, then the DRESP1 or DRESPG responses referenced by the DRESP2 entry must all be of the same analysis type (static, frequency, direct dynamic, modal dynamic, heat transfer or buckling). Static, frequency, heat transfer or buckling responses cannot be mixed. Mass, mass fraction and volume and DRESPG responses cannot be mixed with static, frequency, direct dynamic, modal dynamic, heat transfer or buckling responses.
3. Different static responses **can** be mixed on a single DRESP2 entry. For example, stress and displacement responses can both be referenced by a single DRESP2 entry.
4. DRESP2 can refer to DRESP1's that have multiple physical responses associated with them. However, each referenced DRESP1 has to have the same number of responses. For PTYPE = ELEM, the number of responses is the number of ATTi. For PTYPE = PROP, the number of responses is the sum of the elements associated with each property entered in the referenced DRESP1.
5. For RTYPE = DISP, the number of ATTi may be different on each DRESP1 entry, but the number of ATTi multiplied by the number of components must be the same. DISP responses on the DRESP1 entries can reference different grids and components (directions).
6. DRESP2 entries must have unique identification numbers with respect to DRESP1, DRESP3, DRESPG and DRESPU entries.
7. The sequence in which DVAR, DTABLE, DGRID, and DRESP1 / DRESP1L / DRESP2 occur is not arbitrary. They must be in the order shown. Any of these six words along with the identification numbers associated with them can be omitted if they are not involved in this DRESP2 relationship. However, at least one of the words DVAR, DGRID, DRESP1 / DRESP1L / DRESP2 should exist.

8. DRESP2 entries that reference DRESP1 entries which are mass, mass fraction or volume responses will be in a distinct REGION. Two different DRESP2 entries that reference static, frequency, direct dynamic, modal dynamic, heat transfer and/or buckling responses will never be in the same REGION, even if they are assigned the same REGION identification number. DRESP1 and DRESP2 responses will never be contained in the same region, even if they are assigned the same REGION identification number. The default is to put all responses referenced by one DRESP2 entry in a distinct region.
9. If a DRESP2 entry is not referenced by a DOBJ or DCONS/DCONS2 entry, this response is calculated for all load cases and printed with other DRESP2 values in the DRESP2 response table.
10. The variables identified by NDVi, NCj, NGn, NGCn, NRk and NSk correspond to parameters listed in the left-hand side of the equation on the DEQATN entry identified by EQID. The parameters are assumed to be; first, according to the order of DVAR, DTABLE, DGRID and DRESP1 / DRESP2 specifications, and second, NDVi, NCj, NGn, NGCn, NRk / NSk. In the example above, the order is: DVAR 87, 25, etc.; then DTABLE 1, 8, etc.; then DGRID 2, 3, etc., and finally DRESP1 12, 5, etc.
11. The value identified by NGn and NGCn is the component of the location of the grid in the **basic** coordinate system.
12. If the DRESP1L keyword is used, the responses from different analysis types; static, frequency, direct dynamic, modal dynamic, heat transfer, buckling and/or DRESPG can be mixed.
13. If the DRESP1L keyword is used, then the LID on the DCONS/DCONS2 data must be blank.
14. If the DRESP1L keyword is used, then the LID on the DOBJ data must be the same as the LID of the first DRESP1 referenced (LIDR1).
15. If the DRESP1L keyword is used and the response type is MASS, MASSFR, VOLUME, INERTIA or DRESPG, then the LID should be the ID of the first LOADCASE.
16. If the DRESP1L keyword is used with dynamic responses, then each dynamic loadcase referenced (LIDi) must have the same number of loading frequencies.
17. The name CYCLE can be used in the DTABLE list to pass the current value of the design cycle number (as long as the name CYCLE is not explicitly defined on the DTABLE entry).
18. NDVi, NRk and NSk can be an integer or a string "THRU". If a NDVi is "THRU", it specifies a sequential list of DVAR; if a NRk is "THRU", it specifies a sequential list of DRESP1 or DRESPG; if a NSk is "THRU", it specifies a sequential list of DRESP2 or DRESP3.

8.2.14 DRESP3

Data Entry: **DRESP3** - Create Synthetic Response Quantities.

Description: Define user-subroutine or built-in responses that can be used in the design, either as constraint or as an objective.

Format:

1	2	3	4	5	6	7	8	9	10
DRESP3	ID	LABEL	LIBID or NAME	REGION					
+	"DVAR"	NDV1	NDV2	NDV3	NDV4	NDV5	NDV6	NDV7	NDV8
+		NDV9	-etc.-						
+	"DTABLE"	NC1	NC2	NC3	NC4	NC5	NC6	NC7	NC8
+		NC9	-etc.-						
+	"DGRID"	NG1	NGC1	NG2	NGC2	NG3	NGC3	NG4	NGC4
+		NG5	NGC5	-etc.-					
+	"DRESP1"	NR1	NR2	NR3	NR4	NR5	NR6	NR7	NR8
+		NR9	-etc.-						

Alternate Format 1:

1	2	3	4	5	6	7	8	9	10
DRESP3	ID	LABEL	LIBID or NAME						
+	"DVAR"	NDV1	NDV2	NDV3	NDV4	NDV5	NDV6	NDV7	NDV8
+		NDV9	-etc.-						
+	"DTABLE"	NC1	NC2	NC3	NC4	NC5	NC6	NC7	NC8
+		NC9	-etc.-						
+	"DGRID"	NG1	NGC1	NG2	NGC2	NG3	NGC3	NG4	NGC4
+		NG5	NGC5	-etc.-					
+	"DRESP1L"	NR1	LIDR1	NR2	LIDR2	NR3	LIDR3	NR4	LIDR4
+		NR5	LIDR5	-etc.-					

Alternate Format 2:

1	2	3	4	5	6	7	8	9	10
DRESP3	ID	LABEL	LIBID or NAME						
+	"DVAR"	NDV1	NDV2	NDV3	NDV4	NDV5	NDV6	NDV7	NDV8
+		NDV9	-etc.-						
+	"DTABLE"	NC1	NC2	NC3	NC4	NC5	NC6	NC7	NC8
+		NC9	-etc.-						
+	"DGRID"	NG1	NGC1	NG2	NGC2	NG3	NGC3	NG4	NGC4
+		NG5	NGC5	-etc.-					
+	"DRESP2"	NS1	NS2	NS3	NS4	NS5	NS6	NS7	NS8
+		NS9	-etc.-						

Example: User-Subroutine Name - GNSUB3

1	2	3	4	5	6	7	8	9	10
DRESP3	1	EBUCK	3	4					
+	DVAR	87	25	3	8				
+	DTABLE	VAL1	YOUNGS	CVAL4					
+	DGRID	2	3	4	2				
+	DRESP1	12	5	2	102	202	302	402	502
+		602	702						

Example: Built-in equation AVG3

1	2	3	4	5	6	7	8	9	10
DRESP3	1	SUMM*3	AVG3	4					
+	DVAR	1	2	3	4				
+	DGRID	1001	1	1001	2				
+	DRESP1	10	20	30	40	50	60	70	80
+		90	100						

Field Information Description

- 2 ID Unique identification number. (Integer > 0). See remark 6.
- 3 LABEL User defined label for information only.

DRESP3

Shape, Sizing, Topography, Topometry and Freeform Design Model Data

4	LIBID or NAME	Library identification number. (Integer > 0). or one of the following built-in function names: SUM1, SUM2, SUM3, SUM4, AVG1, AVG2, AVG3, AVG4, NORM1, NORM2, NORM3, NORM4, MULT1, MULT2, MULT3, MULT4, SAVG1, SAVG2, SAVG3, SAVG4, ASAVG1, ASAVG2, ASAVG3, ASAVG4, PNORM2 and SDEV. See remark 18.
5	REGION	Region identifier for constraint screening. (Integer > 0 or blank. Default puts this set of responses in a unique region.).
2	DVAR	Word indicating DVAR identification numbers.
3,4,..	NDVi	DVAR identification number. (Integer > 0 or Blank or "THRU").
2	DTABLE	Word indicating constant identification numbers in DTABLE.
3,4,..	NCj	Name of constant defined in DTABLE input. (Character). See Remark 19.
2	DGRID	Word indicating GRID identification and component numbers. See remark 10.
3,5,..	NGn	GRID identification number (Integer > 0 or Blank).
4,6,..	NGCn	GRID component number ($1 \leq \text{Integer} \leq 3$ or Blank).
2	DRESP1	Word indicating DRESP1 or DRESPG identification numbers.
3,4,..	NRk	DRESP1 or DRESPG identification number. (Integer > 0 or Blank or "THRU"). See remarks 2, 4 and 5.
2	DRESP1L	Word indicating DRESP1 or DRESPG and Loadcase identification numbers.
3,5,..	NRk	DRESP1 identification number (Integer > 0 or Blank). See remark 12.
4,6,..	LIDRk	Loadcase identification number (Integer > 0).
2	DRESP2	Word indicating DRESP2 or DRESP3 identification numbers.
3,4,..	NSk	DRESP2 or DRESP3 identification number. (Integer > 0 or Blank or "THRU").

Remarks:

1. DRESP3 entries can only reference DVAR, DTABLE, DRESP1, DRESP2, DRESP3, DRESPG and GRID entries. They cannot reference DRESPU entries.
2. If the DRESP1 keyword is used, then the DRESP1 or DRESPG responses referenced by the DRESP3 entry must all be of the same analysis type (static, frequency, direct dynamic, modal dynamic, heat transfer or buckling). Static, frequency, heat transfer and buckling responses cannot be mixed. Mass, mass fraction, volume and DRESPG responses cannot be mixed with static, frequency, direct dynamic, modal dynamic, heat transfer or buckling responses.
3. Different static responses **can** be mixed on a single DRESP3 entry. For example, stress and displacement responses can both be referenced by a single DRESP3 entry.

4. DRESP3 can refer to DRESP1's that have multiple physical responses associated with them. However, each referenced DRESP1 has to have the same number of responses. For PTYPE = ELEM, the number of responses is the number of ATTi. For PTYPE = PROP, the number of responses is the sum of the elements associated with each property entered in the referenced DRESP1.
5. For RTYPE = DISP, the number of ATTi may be different on each DRESP1 entry, but the number of ATTi multiplied by the number of components must be the same. DISP responses on the DRESP1 entries can reference different grids and components (directions).
6. **DRESP3** entries must have unique identification numbers with respect to **DRESP1**, **DRESP2**, **DRESPG** and **DRESPU** entries.
7. The sequence in which DVAR, DTABLE, DGRID, and DRESP1 / DRESP1L / DRESP2 occur is not arbitrary. They must be in the order shown. Any of these six words along with the identification numbers associated with them can be omitted if they are not involved in this DRESP3 relationship. However, at least one of the words DVAR, DGRID, DRESP1 / DRESP1L / DRESP2 should exist.
8. DRESP3 entries that reference DRESP1 entries which are mass, mass fraction or volume responses will be in a distinct REGION. Two different DRESP3 entries that reference static, frequency, direct dynamic, modal dynamic heat transfer and/or buckling responses will never be in the same REGION, even if they are assigned the same REGION identification number. DRESP1 and DRESP3 responses will never be contained in the same region, even if they are assigned the same REGION identification number. The default is to put all responses referenced by one DRESP3 entry in a distinct region.
9. If a DRESP3 entry is not referenced by a DOBJ or DCONS/DCONS2 entry, this response is calculated for all load cases and printed with other DRESP3 values in the DRESP3 response table.
10. The variables identified by NDVi, NCj, NGn, NGCn, NRk and NSk correspond to values in the VAR array in the GNSUBi user routine. The parameters are assumed to be; first, according to the order of DVAR, DTABLE, DGRID, and DRESP1 / DRESP2 specifications, and second, NDVi, NCj, NGn, NGCn, NRk / NSk. In the example above, the order is: DVAR 87, 25, etc.; then DTABLE 1, 8, etc.; then DGRID 2, 3, etc., and finally DRESP1 12, 5, etc.
11. The value identified by NGn and NGCn is the component of the location of the grid in the **basic** coordinate system.
12. If the DRESP1L keyword is used, the responses from different analysis types; static, frequency, direct dynamic, modal dynamic, heat transfer, buckling and/or DRESPG, can be mixed.
13. If the DRESP1L keyword is used, then the LID on the DCONS/DCONS2 data must be blank.
14. If the DRESP1L keyword is used, then the LID on the DOBJ data must be the same as the LID of the first DRESP1 referenced (LIDR1).
15. If the DRESP1L keyword is used and the response type is MASS, MASSFR, VOLUME, INERTIA or DRESPG, then the LID should be the ID of the first LOADCASE.

16. If a library ID is given, the **DRESP3** executive control command must be used to identify the shared object (DLL) that contains the user calculation subroutines. See [Use of the DRESP3 / TRESP3 capability](#) (p. 307) for a description of how to create the DRESP3 interface function.
17. If the DRESP1L keyword is used with dynamic responses, then each dynamic loadcase referenced (LID_i) must have the same number of loading frequencies.
18. The built-in functions are defined as follows (X1, ..., XN are all of the DVAR, DTABLE, DGRID, and DRESP1 / DRESP1L / DRESP2 argument values; N is the number of arguments; LBi and UBi are the corresponding lower bound and upper bound from the DVAR entry for DVAR arguments or LBi = 0.0 and UBi = 1.0 for all other arguments types):

$$\begin{aligned} \text{AVG1} &= (X1 + X2 + \dots + XN)/N \\ \text{AVG2} &= (X1^{**2} + X2^{**2} + \dots + XN^{**2})/N \\ \text{AVG3} &= (X1^{**3} + X2^{**3} + \dots + XN^{**3})/N \\ \text{AVG4} &= (X1^{**4} + X2^{**4} + \dots + XN^{**4})/N \end{aligned}$$

$$\begin{aligned} \text{MULT1} &= X1 * X2 * \dots * XN \\ \text{MULT2} &= \text{MULT1}^{**2} \\ \text{MULT3} &= \text{MULT1}^{**3} \\ \text{MULT4} &= \text{MULT1}^{**4} \end{aligned}$$

$$\begin{aligned} \text{NORM1} &= (\text{ABS}(X1) + \text{ABS}(X2) + \dots + \text{ABS}(XN)) \\ \text{NORM2} &= (\text{ABS}(X1)^{**2} + \text{ABS}(X2)^{**2} + \dots + \text{ABS}(XN)^{**2})^{**}(1/2) \\ \text{NORM3} &= (\text{ABS}(X1)^{**3} + \text{ABS}(X2)^{**3} + \dots + \text{ABS}(XN)^{**3})^{**}(1/3) \\ \text{NORM4} &= (\text{ABS}(X1)^{**4} + \text{ABS}(X2)^{**4} + \dots + \text{ABS}(XN)^{**4})^{**}(1/4) \end{aligned}$$

$$\begin{aligned} \text{SUM1} &= X1 + X2 + \dots + XN \\ \text{SUM2} &= X1^{**2} + X2^{**2} + \dots + XN^{**2} \\ \text{SUM3} &= X1^{**3} + X2^{**3} + \dots + XN^{**3} \\ \text{SUM4} &= X1^{**4} + X2^{**4} + \dots + XN^{**4} \end{aligned}$$

$$\begin{aligned} \text{SAVG1} &= ((X1-\text{LB1})/(\text{UB1}-\text{LB1}) + (X2-\text{LB2})/(\text{UB2}-\text{LB2}) + \dots + (XN-\text{LBN})/(\text{UBN}-\text{LBN}))/N \\ \text{SAVG2} &= (((X1-\text{LB1})/(\text{UB1}-\text{LB1}))^{**2} + ((X2-\text{LB2})/(\text{UB2}-\text{LB2}))^{**2} + \dots + (XN-\text{LBN})/(\text{UBN}-\text{LBN}))^{**2})/N \\ \text{SAVG3} &= (((X1-\text{LB1})/(\text{UB1}-\text{LB1}))^{**3} + ((X2-\text{LB2})/(\text{UB2}-\text{LB2}))^{**3} + \dots + (XN-\text{LBN})/(\text{UBN}-\text{LBN}))^{**3})/N \\ \text{SAVG4} &= (((X1-\text{LB1})/(\text{UB1}-\text{LB1}))^{**4} + ((X2-\text{LB2})/(\text{UB2}-\text{LB2}))^{**4} + \dots + (XN-\text{LBN})/(\text{UBN}-\text{LBN}))^{**4})/N \end{aligned}$$

$$\text{ASAVG1} = (\text{ABS}(\text{X1})/\text{MAX}(\text{ABS}(\text{UB1}),\text{ABS}(\text{LB1}))) + \\ \text{ABS}(\text{X2})/\text{MAX}(\text{ABS}(\text{UB2}),\text{ABS}(\text{LB2})) + \dots + \\ \text{ABS}(\text{XN})/\text{MAX}(\text{ABS}(\text{UBN}),\text{ABS}(\text{LBN}))/\text{N}$$

$$\text{ASAVG2} = ((\text{ABS}(\text{X1})/\text{MAX}(\text{ABS}(\text{UB1}),\text{ABS}(\text{LB1}))))^{**2} + \\ (\text{ABS}(\text{X2})/\text{MAX}(\text{ABS}(\text{UB2}),\text{ABS}(\text{LB2}))))^{**2} + \dots + \\ (\text{ABS}(\text{XN})/\text{MAX}(\text{ABS}(\text{UBN}),\text{ABS}(\text{LBN}))))^{**2}/\text{N}$$

$$\text{ASAVG3} = ((\text{ABS}(\text{X1})/\text{MAX}(\text{ABS}(\text{UB1}),\text{ABS}(\text{LB1}))))^{**3} + \\ (\text{ABS}(\text{X2})/\text{MAX}(\text{ABS}(\text{UB2}),\text{ABS}(\text{LB2}))))^{**3} + \dots + \\ (\text{ABS}(\text{XN})/\text{MAX}(\text{ABS}(\text{UBN}),\text{ABS}(\text{LBN}))))^{**3}/\text{N}$$

$$\text{ASAVG4} = ((\text{ABS}(\text{X1})/\text{MAX}(\text{ABS}(\text{UB1}),\text{ABS}(\text{LB1}))))^{**4} + \\ (\text{ABS}(\text{X2})/\text{MAX}(\text{ABS}(\text{UB2}),\text{ABS}(\text{LB2}))))^{**4} + \dots + \\ (\text{ABS}(\text{XN})/\text{MAX}(\text{ABS}(\text{UBN}),\text{ABS}(\text{LBN}))))^{**4}/\text{N}$$

$$\text{PNORM2} = \text{SQRT} (\text{MAX}(\text{X1},0.0)^{**2} + \text{MAX}(\text{X2},0.0)^{**2} + \dots + \\ \text{MAX}(\text{XN},0.0)^{**2})$$

$$\text{SDEV} = \text{SQRT} (((\text{X1}-\text{AVG1})^{**2} + (\text{X2}-\text{AVG1})^{**2} + \dots + (\text{XN}-\text{AVG1})^{**2})/\text{N})$$

19. The name CYCLE can be used in the DTABLE list to pass the current value of the design cycle number (as long as the name CYCLE is not explicitly defined on the DTABLE entry).
20. NDVi, NRk and NSk can be an integer or a string "THRU". If a NDVi is "THRU", it specifies a sequential list of DVAR; if a NRk is "THRU", it specifies a sequential list of DRESP1 or DRESPG; if a NSk is "THRU", it specifies a sequential list of DRESP2 or DRESP3.

8.2.15 DRESPG

Data Entry: **DRESPG**- Select Geometric Response Quantities.

Description: Define a geometric response that it is used either as a constraint or as an objective or as an argument of DRESP2/3.

Format:

1	2	3	4	5	6	7	8	9	10
DRESPG	ID	LABEL	RTYPE				ATT1	ATT2	ATT3
+	ATT4	-etc.-							

Example:

1	2	3	4	5	6	7	8	9	10
DRESPG	101	LENGTH1	LENGTH				10	11	12

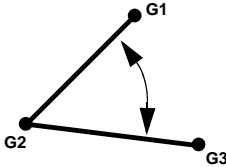
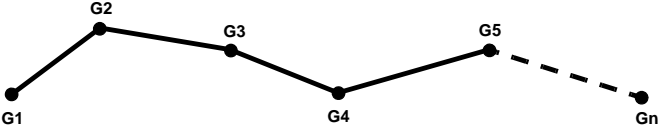
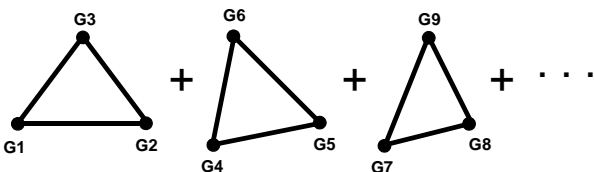
Field Information Description

2	ID	Unique identifier (Integer > 0). See Remark 1.
3	LABEL	User defined label for information only (Characters or blank).
4	RTYPE	Response type: ANGLE, LENGTH, AREA3, AREA4, VOL4, VOL6, VOL8, DGLINE, DGPLANE, DIFFX, DIFFY, DIFFZ, DISTX, DISTY or DISTZ.
8,9,..	ATTi	Grid ID's (Integer > 0).

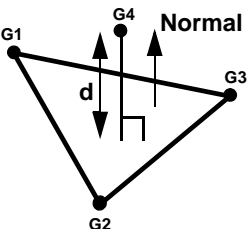
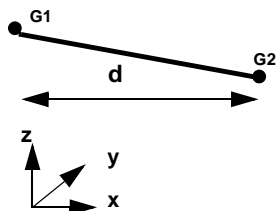
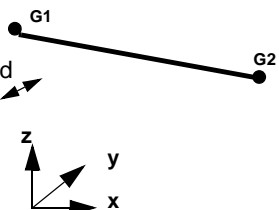
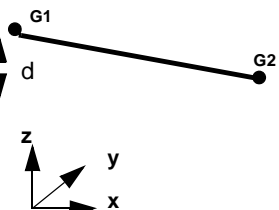
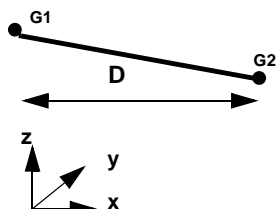
Remarks:

1. **DRESPG** identification numbers must be unique with respect to **DRESP1**, **DRESP2**, **DRESP3** and **DRESPU** identification numbers.
2. For RTYPE = LENGTH, the response is the sum of the distances between consecutive grids in the ATTi list.
3. For RTYPE = ANGLE, the response is the angle between the line defined by GID₂ and GID₁ and the line defined by GID₂ and GID₃.
4. For RTYPE = AREA3 or AREA4, the response is the sum of the areas defined by the grids.
5. For RTYPE = VOL4, VOL6 or VOL8, the response is the sum of the volumes defined by the grids.
6. For RTYPE = DGLINE, the response is the distance between the GID3 and the line defined by GID1 and GID2

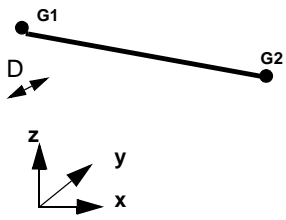
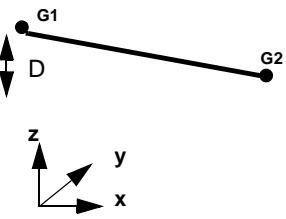
7. For **RTYPE = DGPLANE**, the response is the distance between **GID4** and the plane defined by **GID1**, **GID2** and **GID3**. If the grid **GID4** is on the positive side of the plane, then the distance will be positive. The positive side of the plane is the side where the normal vector of the plane points. The normal of the plane is constructed using **GID1**, **GID2** and **GID3**. See the figure below.
8. For **RTYPE = DIFFX**, the response is the difference between the x coordinate of **G2** and the x coordinate of **G1**. The coordinates are in the basic coordinate system. This response can be positive, zero or negative.
9. For **RTYPE = DIFFY**, the response is the difference between the y coordinate of **G2** and the y coordinate of **G1**. The coordinates are in the basic coordinate system. This response can be positive, zero or negative.
10. For **RTYPE = DIFFZ**, the response is the difference between the z coordinate of **G2** and the z coordinate of **G1**. The coordinates are in the basic coordinate system. This response can be positive, zero or negative.
11. For **RTYPE = DISTX**, the response is the absolute values of the difference between the x coordinate of **G2** and the x coordinated of **G1**. The coordinates are in the basic coordinate system. This response can only be positive or zero.
12. For **RTYPE = DISTY**, the response is the absolute values of the difference between the y coordinate of **G2** and the y coordinated of **G1**. The coordinates are in the basic coordinate system. This response can only be positive or zero.
13. For **RTYPE = DISTZ**, the response is the absolute values of the difference between the y coordinate of **G2** and the z coordinated of **G1**. The coordinates are in the basic coordinate system. This response can only be positive or zero.

RTYPE	DEFINITION (Gi used here is the grid ID, GIDi)
ANGLE	
LENGTH	
AREA3	

AREA4	
VOL4	
VOL6	
RTYPE	DEFINITION (Gi used here is the grid ID, GIDi)
VOL8	
DGLINE	

DGPLANE	 <p>IF GID4 IS ON THE POSITIVE SIDE OF THE PLANE, $d > 0$</p> <p>IF GID4 IS ON THE NEGATIVE SIDE OF THE PLANE, $d < 0$</p>
DIFFX	 <p>IF GID2 IS ON THE POSITIVE X SIDE OF THE GID1, $d > 0$</p> <p>IF GID2 IS ON THE NEGATIVE X SIDE OF GRID GID1, $d < 0$</p> <p>d is measured in the basic coordinate system</p>
DIFFY	 <p>IF GID2 IS ON THE POSITIVE Y SIDE OF THE GID1, $d > 0$</p> <p>IF GID2 IS ON THE NEGATIVE Y SIDE OF GRID GID1, $d < 0$</p> <p>d is measured in the basic coordinate system</p>
DIFFZ	 <p>IF GID2 IS ON THE POSITIVE Z SIDE OF THE GID1, $d > 0$</p> <p>IF GID2 IS ON THE NEGATIVE Z SIDE OF GRID GID1, $d < 0$</p> <p>d is measured in the basic coordinate system</p>
DISTX	 <p>D is always positvel or zero (DISTX ≥ 0)</p> <p>D is measured in the basic coordinate system</p>

DRESPG Shape, Sizing, Topography, Topometry and Freeform Design Model Data

DISTY	 <p>D is always positive or zero (DISTY >= 0)</p> <p>D is measured in the basic coordinate system</p>
DISTZ	 <p>D is always positive or zero (DISTZ >= 0)</p> <p>D is measured in the basic coordinate system</p>

8.2.16 DRESPU

Data Entry: **DRESPU**- Identify and Constrain Responses Provided by an External Program.

Description: Defines information related to quantities that are supplied by an external program for use in the optimization phase. DRESPU can be used to define either the objective function or a constraint.

Format:

1	2	3	4	5	6	7	8	9	10
DRESPU	ID	LABEL	LBOUND	UBOUND					

Example:

1	2	3	4	5	6	7	8	9	10
DRESPU	31	OILTEMP	0.0	10.0					

Field	Information	Description
2	ID	Unique identifier (Integer > 0).
3	LABEL	User defined label for information only (Characters or blank).
4	LBOUND	Constraint lower bound imposed on this response (Real or blank. Default = -1.0e30).
5	UBOUND	Constraint upper bound imposed on this response (Real or blank. Default = 1.0e30).

Remarks:

1. The ID must be unique with respect to other **DRESPG**, **DRESPU**, **DRESP1**, **DRESP2** and **DRESP3** IDs.
2. The screening truncation threshold (TRS) for DRESP2 responses (EQUA) will be used.
3. $LBOUND \leq UBOUND$.
4. The number of DRESPU data statements in the bulk data should be the same as the number of responses created by the GNUSER external program. If the number of DRESPU data statements exceeds the number of responses generated by GNUSER a data management error will result and the program will stop.
5. For information regarding the GNUSER external program, see **Use of the DRESPU Capability** (p. 299).
6. DOPT parameters **FDCHU** and **FDCHMU** can be used to select the step size for finite-difference gradient calculations.

8.2.17 DSCREEN

Data Entry: **DSCREEN** - Constraint Screening Data.

Description: Define constraint screening data for constraint deletion.

Format:

1	2	3	4	5	6	7	8	9	10
DSCREEN	TYPE	TRS	NSTR						

Example:

1	2	3	4	5	6	7	8	9	10
DSCREEN	STRESS	-0.5	3						

Field Information Description

2	TYPE	Type of the constraints for which the screening criteria apply. Must be FORCE, STRESS, STRAIN, DISP, TEMP, EQUA, DRESP, MASS, VOLUME, INERTIA, LAMA, SENERGY, FREQ, or RFREQ.
3	TRS	Truncation threshold. (Real \leq 0.0 or blank, Default = -0.5, except for DRESPG where Default = -100.0).
4	NSTR	Maximum number of constraints to be retained per region per load case (per loading frequency). (Integer > 0 or blank) (Default = 20).

Remarks:

1. Displacement constraints associated with each load case are regionalized by the specification on the **DRESP1** data. From each region, up to NSTR displacement constraints are retained per load case (per loading frequency).
2. A particular stress, strain, or force constraint specification may be applied to many elements in a region, generating many stress constraints, but only up to NSTR constraints per load case (per loading frequency) will be retained.
3. If a certain type of constraint exists but no corresponding DSCREEN data is specified, all the screening criteria used for this type of constraint will be furnished by the default value.
4. For GSTRESS, DGSTRESS, CPRESS and FINDEX, use STRESS constraint screening data.

5. Constraints are retained if

$$\frac{\text{Response} - \text{UBi}}{|\text{UBi}|} \geq \text{TRS}$$

or

$$\frac{\text{LBi} - \text{Response}}{|\text{LBi}|} \geq \text{TRS}$$

where Response is defined on the DRESPi entry and LBi and UBi are defined on the DCONS entry or calculated using DCONS2. If |UBi| is zero, the above equation containing |UBi| is replaced by $\text{Response} \geq \text{TRS}$. Similarly, when |LBi| is zero, the equation containing |LBi| is replaced by $\text{Response} \leq -\text{TRS}$.

6. For **DRESP2**, **DRESP3** and **DRESPU**, the screening parameters TRS and NSTR for TYPE = EQUA responses are used.
7. Only one DSCREEN data statement per TYPE is allowed in the bulk data. Thus, the maximum number of DSCREEN data statements in the bulk data is seven.
8. For DSTRESS, DSTRS, PSDSTR or RMSTR use STRESS constraint screening data, for DSTRAIN or DSTNS use STRAIN constraint screening data, and for DFORCE, DFORCES or SPCFORCE use FORCE constraint screening data.
9. For DDISP, DVELO, DACCE, DDISPS, DVELOs, DACCES, MDISP, MVELO, MACCE, MDISPS, MVELOs, MACCES, PSDDISP, PSDVELO, PSDACCE, PSDDS, PSDVS, PSDAS, RMSDISP, RMSVELO, RMSACCE, EVECT and ERP, use DISP constraint screening data.
10. For HTC use SENERGY constraint screening data.
11. The DOPT parameter **FREQREG** can be used to control how responses from different loading frequencies in a frequency response loadcase are grouped into regions.

8.2.18 DSELECT

Data Entry: **DSELECT** - Select a fraction of listed design variables to keep.

Description: Create constraints such that a given fraction of the total number of listed design variables move to their upper bound while the rest of the listed design variables move to their lower bound.

Format:

1	2	3	4	5	6	7	8	9	10
DSELECT	ID	LABEL	FRACT	BTYPE	GTYPE	TOL			
+	"DVAR"	NDV1	NDV2	NDV3	NDV4	NDV5	NDV6	NDV7	NDV8
+		NDV9	-etc.-						

Example 1: Create constraints so that 50% of the listed design variable moves to their upper bound and 50% move to the lower bound.:

1	2	3	4	5	6	7	8	9	10
DSELECT	1	WeldFrac	0.5	EXACT	AVG				
+	DVAR	87	25	3	8				

Example 2: Create constraints so that the average sum of all listed normalized design variables do not exceed 50% of their upper bounds

1	2	3	4	5	6	7	8	9	10
DSELECT	1	WeldFrac	0.5	UPPER	AVG				
+	DVAR	87	25	3	8				

Field Information Description

2	ID	Unique DSELECT identification number. (Integer > 0).
3	LABEL	User defined label for information only.
4	FRACT	Maximum or Exact average of design variable values. (0.0<Real <=1.0)
5	BTYPE	One of the following types: UPPER or EXACT or blank. Default = EXACT. See remark 1.
6	GTYPE	One of the following types: AVG or ABS or blank. Default = AVG. See remark 2.
7	TOL	Tolerance for satisfying selection constraints. Default = 0.005.
2	DVAR	Word indicating DVAR identification numbers.
3,4,..	NDVi	DVAR identification number. (Integer > 0 or "THRU").

Remarks:

1. A DSELECT entry with the BTYPE=EXACT option creates two internal constraints using the listed design variable entries so that a given fraction of them move to their upper bound (GTYPE=AVG) or either bound (GTYPE = ABS) while moving the rest to their lower bound or 0.0. The constraint types are controlled by GTYPE. FRACT controls the fraction of the number of the design variables that are desired to move to their upper bound. A DSELECT entry with the BTYPE=UPPER option creates one internal constraint using the listed design variable entries so that a special average of them does not exceed a given fraction value.
2. If GTYPE = AVG or blank, the following function is used:

$$G = ((X1-LB1)/(UB1-LB1) + (X2-LB2)/(UB2-LB2) + \dots + (XN-LBN)/(UBN-LBN))/N$$
 If GTYPE = ABS, the following function is used:

$$G = (ABS(X1)/MAX(ABS(UB1),ABS(LB1)) + ABS(X2)/MAX(ABS(UB2),ABS(LB2)) + \dots + ABS(XN)/MAX(ABS(UBN),ABS(LBN)))/N$$
3. If BTYPE = UPPER, the following constraint is used:

$$G \leq FRACT + TOL$$
 If BTYPE = EXACT, two constraints will be internally constructed in *GENESIS* so that.

$$FRACT - TOL \leq G \leq FRACT + TOL$$
4. NDVi can be an integer or a string "THRU". If a NDVi is "THRU", it specifies a sequential list of DVAR.

8.2.19 DSHAPE

Data Entry: **DSHAPE** - Attributes of Perturbation Vectors.

Description: Parameters to control attributes of a perturbation vector and/or to define a freeform shape optimization region.

Format:

1	2	3	4	5	6	7	8	9	10
DSHAPE	DVARID	LABEL	SPLIT	FTYPE	MAXPERT	RINIT	SCALE		
+	"SYM"	CID	TYPE1	TYPE2	TYPE3	n	SYMTOL		
+	"COARSE"	CTYPE	DIMEN1						
+	"GRIDFR"	FRACT	BTYPE	GFRTOL					

Example 1: Split (make freeform) all perturbations associated to design variable 100.

1	2	3	4	5	6	7	8	9	10
DSHAPE	100	Top	FREE	DVGRID					

Example 2: Split (make freeform) all perturbations associated to design variable 200. .

1	2	3	4	5	6	7	8	9	10
DSHAPE	200		FREE						

Example 3: Split (make freeform) all perturbations associated to design variable 300. The maximum magnitude of the grid movements associated to any perturbation in the region should be less or equal 5.0.

1	2	3	4	5	6	7	8	9	10
DSHAPE	300	Side	FREE	DVGRID	5.0				

Example 4: Split (make freeform) all perturbations associated to design variable 400. The maximum magnitude of the grid movements associated to any perturbation in the region should be less or equal 5.0. Use mirror symmetries about planes XY and YZ with respect to coordinate system 2.

1	2	3	4	5	6	7	8	9	10
DSHAPE	400	Side	FREE	DVGRID	5.0				
+	SYM	2	MX	MY					

Example 5: Scale by 2.0 all perturbations associated to design variable 500. Keep the perturbation linked (not freeform).

1	2	3	4	5	6	7	8	9	10
DSHAPE	500	Bottom	LINKED	DVGRID			2.0		

Field	Information	Description
2	DVARID	Design variable identification number (Integer>0). Together with its associated DVGRID or DVGRIDC , defines the perturbation vector for which this entry sets attributes. See remark 4 and 5.
3	LABEL	User defined name (Character or blank).
4	SPLIT	Freedom type. One of the words: "FREE" or "LINKED" or blank. Default is "LINKED". See remarks 6 and 7.
5	FTYPE	Designable region type. One of the words: "DVGRIDC", "DVGRID" or blank. Default is "DVGRID". This field is only used if SPLIT is "FREE". See remarks 8 and 9.
6	MAXPERT	Maximum allowed movement of any grid in the direction of the perturbation (Real > 0.0 or blank). See remark 10.
7	RINIT	Parameter to set design variable initial randomness (Real \geq 0.0 or blank. Default=0.0). A value of 0.0 means that no initial randomness will be used. See remarks 11 and 12.
8	SCALE	Scale factor value used to accelerate (SCALE>1.0) or reduce (SCALE<1.0) the speed of convergence of the optimization process. This value will be use to multiply all perturbations in the region and divide the bound of the design variable.(Real > 0.0 or blank. Default = 1.0). See remark 13.
2	SYM	Word indicating that the following data correspond to symmetry data. See remarks 14-17.
3	CID	Coordinate system identification number (Integer \geq 0 or blank. Default = 0)
4	TYPE1	Fabrication constraint type. One of "MYZ", "MZX", "MXY", "CX", "CY", "CZ", "EX", "EY" or "EZ".
5,6	TYPE2, TYPE3	Fabrication constraint type. One of "MYZ", "MZX", "MXY", "CX", "CY", "CZ", "EX", "EY", "EZ" or Blank.
7	n	Number of cyclic symmetry sections. (Integer > =0 or Blank). If n=0 then axisymmetry is used. The value n is only used for "CX", "CY" or "CZ".
8	SYMTOL	Tolerance for checking fabrication symmetry constraints in DSHAPE. Default 0.001.
3	COARSE	Word indicating that coarse parameters will follow. See remark 14 and 18
3	CTYPE	One of the following words: "DIAM" or "NONE". Default is "DIAM". "DIAM" stands for diameter.
5	DIMEN1	Distance for coarsening (Real \geq 0.0 or blank. Default=0.0).
2	GRIDFR	Key word indicating grid movement fraction constraint information will follow. See remarks 14, 19 and 20.

DSHAPE Shape, Sizing, Topography, Topometry and Freeform Design Model Data

3	FRACT	Bead fraction. (0.0<Real <=1.0)
4	BTYPE	One of the following types: "LOWER", "UPPER" or "BOTH". "UPPER" is to get an upper bound on the grid fraction constraint, "LOWER" is to get a lower bound on the bead fraction constraint. "BOTH" is to get a lower and an upper bound on the grid fraction constraint. Default is "UPPER".
5	GFRTOL	Tolerance for satisfying grid fraction constraints. Default 0.005.

Remarks.

1. DSHAPE data is primarily used to split perturbation vectors and to automatically create corresponding design variables. Alternatively, DSHAPE entries can be used to change certain attributes to an existing perturbation vector or its corresponding design variable initial value.
2. Multiple DSHAPE entries are allowed in the input data.
3. DSHAPE data can be used with any other design data
4. Two different DSHAPE entries cannot reference the same design variable.
5. The design variable listed in DVARID must be an independent design variable. That is, DVARID cannot be made dependent by **DLINK**. The design variable can be continuous or discrete.
6. If the SPLIT field is "FREE", perturbations associated to design variable DVARID will be split as directed by the FTYPE field and any given continuation data. Perturbations are said to be "split" when they are altered to reference new independent copies of the design variable.
7. If the SPLIT field is "LINKED" or blank, all perturbations associated to design variable DVARID will be kept associated to the same single design variable. In this case, all continuation lines will be ignored.
8. When FTYPE is "DVGRIDC", the program will split only the DVGRIDC data associated to design variable DVARID. With this option, the DVGRID data generated from each newly split DVGRIDC (and the applicable DOMAIN data) will remain linked. Additional information on FTYPE=DVGRIC can be found in the following chapter: **Freeform with DVGRIDC data** (p. 198).
9. When FTYPE is "DVGRID", the program will split perturbations associated to user supplied DVGRID data as well as DVGRID data automatically generated from DVGRIDC/DOMAIN data. That is, all resulting perturbations associated to DVARID will be split. Additional information on FTYPE=DVGRID can be found in the following chapter: **Freeform with DVGRID data** (p. 196)
10. If MAXPERT is not blank then all perturbations associated to design variable DVARID are scaled such that the largest perturbation (in magnitude) times the largest bound on the design variable (in absolute value) is equal to the MAXPERT value. If MAXPERT is blank, then the perturbations are not changed.

11. If RINIT is 0.0, the program uses the initial design variable value given on the **DVAR** entry for design variable DVARID. A non-zero RINIT value scales the range about the given initial value from which a random value is selected. Larger values of RINIT make it more likely that the initial value will be farther from the initial value given on the DVAR entry. When initial randomness is desired, RINIT=1.0 is recommended.
12. Random initial values are generated by a pseudo-random number generator that gives a repeatable, but system-dependent, sequence of values. Changing the value of the analysis parameter RANDOM will cause the generator to use a different sequence and will lead to different random initial values.
13. The scale factor value (SCALE) is used to either increase (SCALE>1.0) or reduce (SCALE < 1.0) the sensitivities of the responses to the shape change as seen by the optimizer, by scaling the perturbations with the SCALE value. The design variables bounds are scaled with the inverse of SCALE so the the maximum allowed grid perturbation is not altered.
14. All continuation lines are optional. The continuation entries with key words may be enter in any order. Each continuation keyword may used at most once in a single DSHAPE entry. Continuation entries are only used if SPLIT is "FREE".
15. Up to three fabrication constraints are allowed per DSHAPE entry. Following is a table with available fabrication constraints.

TYPE	Description of Fabrication Constraints
MXY	Mirror symmetry with respect to the XY plane
MYZ	Mirror symmetry with respect to the YZ pane
MZX	Mirror symmetry with respect to the ZX plane
CX	Cyclic symmetry about the X axis (n>0) or Axisymmetry about the X axis (n=0)
CY	Cyclic symmetry about the Y axis (n>0) or Axisymmetry about the Y axis (n=0)
CZ	Cyclic symmetry about the Z axis (n>0) or Axisymmetry about the Z axis (n=0)
EX	Extrusion along the X axis
EY	Extrusion along the Y axis
EZ	Extrusion along the Z axis
UXY	Uniform with respect to the XY plane

UYZ	Uniform with respect to the YZ plane
UZX	Uniform with respect to the ZX plane
UXYZ	Uniform in all directions

16. Not all fabrication constraints can be mixed together. Following is a table with the allowed combinations:

<p> MX+MYZ, MX+MZ, MYZ+MZ, MX+MYZ+MZ CX+MYZ, CY+MZ, CZ+MX </p>
<p> EX+MX, EX+MZ, EX+MX+MZ EY+MYZ, EY+MX, EY+MYZ+MX EZ+MZ, EZ+MYZ, EZ+MZ+MYZ </p>
<p>EX+CX, EY+CY, EZ+CZ</p>
<p>UYZ+MYZ, UZX+MZ, UXY+MX</p>

17. For more details on fabrication constraints, see the following chapter: [Defining Fabrication \(Symmetry\) Constraints for Freeform](#) (p. 200)
18. For details on coarse option, refer to the following chapter: [Coarse Freeform](#) (p. 203)
19. The GRIDFR continuation data is optional. If GRIDFR is not used, no constraints will be created.
20. The GRIDFR (Grid fraction) option usually produces more discrete grid changes. The FRACT value, or grid fraction value, represents the fraction of the total number of grids in the region that move from the initial design. Using the BOTH option with FRACT=0.4 will produce an answer in which approximately 40% of the designable grids move to the maximum allowable height, while the other 60% do not move. Using the UPPER option with FRACT=0.4 will produce an answer in which the total grid movement is no more than 40% times the maximum allowable movement times the number of grids in the region. Using the LOWER option with FRACT=0.4 will produce an answer in which the total grid movement is at least 40% times the maximum allowable movement times the number of grids in the region. The UPPER option is recommended for most situations. The LOWER or BOTH options can be used to force the optimizer move the grids in cases where there are low sensitivity values (which can occur in flat structures) and where other options (increasing RINIT value or SCALE) do not help. For more details on this capability, refer to the following chapter [Variability Control - Using the GRIDFR Option in Freeform](#) (p. 204)

8.2.20 DSHIFT

Data Entry: **DSHIFT** - Frequency Response Shifting Definition.

Description: Defines a transformation to shift or scale dynamic displacement, velocity or acceleration responses by a function of the loading frequency.

Format:

1	2	3	4	5	6	7	8	9	10
DSHIFT	ID	COEFF	FTYPE	REFVAL	TID	ETYPE			

Example 1: Create a shifted response, where the tabled values are used to normalize the response. Assume that a DRESP1 that reference MACCES points to a DSHIFT 1001

$$S(f) = \frac{20 \log_{10} \left(\frac{ACCE(f)}{0.02} \right)}{T(f)} \quad \therefore$$

1	2	3	4	5	6	7	8	9	10
DSHIFT	1001	20.0	LOG10	0.02	20001	2			

Example 2: Create a shifted response, where the tabled values are subtracted from the response. Assume that a DRESP1 that references MACCES points to DSHIFT 1002

$$.S(f) = 20 \log_{10} \left(\frac{ACCE(f)}{0.02} \right) - T(f) :$$

1	2	3	4	5	6	7	8	9	10
DSHIFT	1002	20.0	LOG10	0.02	20001	1			

Format:

Field Information Description

2	ID	DSHIFT identification number (Integer > 0).
3	COEFF	Scale factor (Real or blank. Default = 1.0).
4	FTYPE	Function type. One of "IDENT", "LOG10", "LOG", "RECIP" or Blank. Default is "IDENT". See remark 3.
5	REFVAL	Reference value. This value divides the frequency response H(f) associated to the DRESP1 that references DSHIFT. (Real > 0.0 or blank. Default = 1.0).
6	TID	Identification number of a TABLED1, TABLED2, TABLED3 or TABLED4 entry which defines T(f) (Integer > 0).
7	ETYPE	Equation type. (Integer 1 or 2). If ETYPE = 1: TABLEDi values are subtracted. If ETYPE = 2: TABLEDi values are used for normalization.

Remarks:

1. DSHIFT entries are used to specify parameters to create customized responses from frequency response loadcases that are transformed by a user-specified function of the loading frequency. Shifted responses are function of the dynamic displacement, velocities and accelerations that are listed in **DRESP1**, and are created by listing DDISPS, DVELOs, DACCES, MDISPS, MVELOs, MACCES, PSDDS, PSDVS, PSDAS, UFDISPS, UFVELOs, UFACCES, ERPS, DSTRS, DSTNS or DFORCES in DRESP1 and referencing the DSHIFT data.
2. Shifted responses can be used to create a design constraint in which the bound is a function of the loading frequency. See **Loading Frequency Dependent Constraint Bounds** (p. 320)
3. In the following tables, $H(f)$ represents the dynamic displacement, velocity, acceleration or element result magnitude specified on the referencing DRESP1. For FTYPE = IDENT, the following transformations are used:

	FTYPE = IDENT
ETYPE = 1	$S(f) = \text{Coeff} \times \frac{H(f)}{\text{Refval}} - T(f)$
ETYPE = 2	$S(f) = \frac{\text{Coeff} \times \frac{H(f)}{\text{Refval}}}{T(f)}$

For FTYPE = LOG10, the following transformations are used

	FTYPE = LOG10
ETYPE = 1	$S(f) = \text{Coeff} \times \log_{10}\left(\frac{H(f)}{\text{Refval}}\right) - T(f)$
ETYPE = 2	$S(f) = \frac{\text{Coeff} \times \log_{10}\left(\frac{H(f)}{\text{Refval}}\right)}{T(f)}$

For FTYPE = LOG, the following transformations are used

	FTYPE = LOG
ETYPE = 1	$S(f) = \text{Coeff} \times \ln\left(\frac{H(f)}{\text{Refval}}\right) - T(f)$
ETYPE = 2	$S(f) = \frac{\text{Coeff} \times \ln\left(\frac{H(f)}{\text{Refval}}\right)}{T(f)}$

For FTYPE = RECIP, the following transformations are used:

	FTYPE = RECIP
ETYPE = 1	$S(f) = \text{Coeff} \times \frac{\text{Refval}}{H(f)} - T(f)$
ETYPE = 2	$S(f) = \frac{\text{Coeff} \times \frac{\text{Refval}}{H(f)}}{T(f)}$

8.2.21 DSPLIT

Data Entry: **DSPLIT** - Topometry Data Entry.

Description: Defines a region for topometry optimization.

Format:

1	2	3	4	5	6	7	8	9	10
DSPLIT	ID	LABEL	PTYPE	PID					
+	"PLINK"	PID1	PID2	PID3	PID4	PID5	PID6	PID7	PID8
+		PID9	PID10	-etc.-					
+	"COARSE"	CTYPE	CVALUE1	CVALUE2	CVALUE3				
+	"SYM"	CID	TYPE1	TYPE2	TYPE3	n	SYMTOL	SYMMET	ANGTYPE
+	"NDVAR"	DV1	DV2	DV3	DV4	DV5	DV6	DV7	DV8
+		DV9	DV10	-etc.-					

Alternate Format:

1	2	3	4	5	6	7	8	9	10
DSPLIT	ID	LABEL	PTYPE	PID					
+	"PLINK"	PID1	PID2	PID3	PID4	PID5	PID6	PID7	PID8
+		PID9	PID10	-etc.-					
+	"COARSE"	CTYPE	CVALUE1	CVALUE2	CVALUE3				
+	"SYM"	CID	TYPE1	TYPE2	TYPE3	n	SYMTOL	SYMMET	ANGTYPE
+	"DVAR"	DV1	DV2	DV3	DV4	DV5	DV6	DV7	DV8
+		DV9	DV10	-etc.-					

Example 1: Split elements that reference PSHELL 100.

1	2	3	4	5	6	7	8	9	10
DSPLIT	1012	TOPSURF	PSHELL	100					

Example 2: Split elements that reference PBAR 22.

1	2	3	4	5	6	7	8	9	10
DSPLIT	221								

Example 3: Split elements that reference PHELL 100, use coarse symmetry grouping 4 elements per design variable.,

1	2	3	4	5	6	7	8	9	10
DSPLIT	100								
+	COARSE	MAXELEM	4						

Example 4: Split elements that reference PHELL 200, use coarse symmetry grouping 2 elements per design variable, and use triple mirror symmetries.,

1	2	3	4	5	6	7	8	9	10
DSPLIT	100		PSHELL	200					
+	COARSE	MAXELEM	2						
+	SYM	0	MXV	MYZ	MZX				

Example 5: Split elements that reference PBAR 221, do not split DVAR 10.,

1	2	3	4	5	6	7	8	9	10
DSPLIT	102	LATERAL	PBAR	221					
+	NDVAR	10							

Field Information Description

2	ID	Unique DSPLIT region identification number. (Integer > 0).
3	LABEL	User defined name for output purposes (Character or blank).
4	PTYPE	Designable region type. One of the following words: PROD, PBAR, PSHELL, PCOMP, PSHEAR, PELAS, PELASH, PCONM3, PHBDY, PDAMP, PMASS, PVISC, PAXIS, PVECTOR, PBUSH or PROP (Default = PROP).
5	PID	Property identification number (Integer>0 or Blank. Default = ID). See Remark 2.
2	PLINK	Word indicating that the following data correspond to property identification numbers. This data is optional. See Remark 3.
5	PID1	Property identification numbers (Integer>0). See Remark 3.
5	PID2...PIDn	Property identification numbers (Integer>0 or Blank). See Remark 3.
2	COARSE	Word indicating that the following data correspond to coarse topometry data. This data is optional. See Remark 4 and 21.
3	CTYPE	Coarsening type. MAXELEM or LENGTH. See Remark 4.
4,5,6	CVALUEi	For CTYPE=MAXELEM, CVALUE1 is the desired maximum number of elements per group (Integer>0). For CTYPE=LENGTH, CVALUE1 is the size in the x-direction, CVALUE2 is the size in the y-direction and CVALUE3 is the size in the z-direction (Real > 0.0).
2	SYM	Word indicating that the following data correspond to symmetry topometry data. This data is optional. See Remark 6, 20 and 21.
3	CID	Coordinate system identification number (Integer \geq 0 or blank. Default = 0)

DSPLIT

Shape, Sizing, Topography, Topometry and Freeform Design Model Data

4,5,6	TYPEi	Fabrication constraint type. One of "MYZ", "MZX", "MXY", "CX", "CY", "CZ", "EX", "EY", "EZ", "UYZ", "UZX", "UXY", "UXYZ" or Blank. See Remarks 6 and 7.
7	n	Number of cyclic symmetry sections. (Integer > 1 or Blank).
8	SYMTOL	Tolerance for checking fabrication symmetry constraints in DSPLIT. Default 0.001.
9	SYMMET	Symmetry validation method for DSPLIT. 1: Center location 2: Corner grids of elements and center locations (Default).
10	ANGTYPE	Treatment of symmetry for PCOMP layer angles. "ANGREP": Repeat angles. Angles of symmetrical elements are forced to have the same magnitudes and the same signs. "ANGSYM": Mirror Angles: Angles of symmetrical elements are forced to have the same magnitudes but opposite signs (Default).
2	NDVAR	Word indicating that the following DVAR IDs are not to be split. This list is optional. See Remark 9.
3, 4, ...	DVi	DVAR entry identification number (Integer > 0 or Blank).
2	DVAR	Word indicating that only the following DVAR IDs are to be split. This list is optional. See Remark 9.
3, 4, ...	DVi	DVAR entry identification number (Integer > 0 or Blank).

Remarks.

1. Multiple DSPLIT entries are allowed in the input data, but each DSPLIT must reference a different PID.
2. DSPLIT data is used to perform element-by-element sizing optimization. To achieve this, the program automatically replicates property PID such that all elements that reference PID have their own independent copy. The program automatically generates new design data (DVPROPi and possibly DVAR) associated to the new internally created properties to match existing design data associated to PID. All real numbers on automatically generated data have same values as the corresponding numbers on the original data, so an initial design cycle with or without DSPLIT would generate the same answers.
3. The optional PLINK list is used to create an "extended topometry" region. Extended topometry regions are primarily used to enforce symmetries across different properties. Property PID is called the master property. The properties that are listed in the PLINK list are called slave properties. Slave properties can only be listed on one DSPLIT entry. Slave properties have to be of the same type as the master properties (a PSHELL master can not have a PCOMP as a slave property). Slave properties must not have sizing data. The property values of a slave property are ignored by the program. The property values used by the program are from the master property.

4. DSPLIT data can also be used to perform “Coarse Topometry” or group-by-group sizing optimization. With the COARSE continuation data, the program automatically generates groups of elements to be designed with the same set of design variables. For CTYPE= MAXELEM the groups will be contiguous and CVALUE1 defines the desired number of elements in each group. Some generated groups may contain fewer elements than the desired number. The larger the number of elements in a group, the smaller the number of groups in a given topometry region, and therefore, the smaller the number of generated design variables. For CTYPE=LENGTH, space is divided into a uniform mesh of boxes, each of dimension CVALUE1 by CVALUE2 by CVALUE3. Elements whose centroids are within a given box are grouped together. The boxes are aligned with the axes of the coordinate system identified by CID on the SYM continuation line.
5. The COARSE option allows to trade numbers of design variables for speed and computer resources (memory and disk space). A large value of MAXELEM can be used to reproduce the sizing, while a value of 1 can be used to reproduce standard topometry results. Note that when using a large value of MAXELEM to reproduce sizing, if the sizing region contains disjoint parts, coarse topometry will separate them into separate design regions.
6. Up to three fabrication constraints are allowed per DSPLIT entry. Following is a table with available fabrication constraints.

TYPE	Description of Fabrication Constraints
MXY	Mirror symmetry with respect to the XY plane
MYZ	Mirror symmetry with respect to the YZ pane
MZX	Mirror symmetry with respect to the ZX plane
CX	Cyclic symmetry about the X axis
CY	Cyclic symmetry about the Y axis
CZ	Cyclic symmetry about the Z axis
EX	Extrusion along the X axis
EY	Extrusion along the Y axis
EZ	Extrusion along the Z axis
UXY	Uniform in planes parallel to the XY plane
UYZ	Uniform in planes parallel to the YZ pane
UZX	Uniform in planes parallel to the ZX plane
UXYZ	Uniform in all directions

- Not all fabrication constraints can be mixed together. Following is a table with the allowed combinations:

$ \begin{aligned} & \text{MXY+MYZ, MXY+MZX, MYZ+MZX, MXY+MYZ+MZX} \\ & \text{CX+MYZ, CY+MZX, CZ+MXY} \\ & \text{EX+MXY, EX+MZX, EX+MXY+MZX} \\ & \text{EY+MYZ, EY+MXY, EY+MYZ+MXY} \\ & \text{EZ+MZX, EZ+MYZ, EZ+MZX+MYZ} \\ & \text{EX+CX, EY+CY, EZ+CZ} \\ & \text{UYZ+MYZ, UZX+MZX, UXY+MXY} \end{aligned} $

- Fabrication constraints and coarse topometry can be used simultaneously. In this case, the number of elements designed by the same set of variables will be typically larger than MAXELEM. For example, using 2 elements per group and mirror symmetry will produce design variables that design 4 elements.
- A DVAR is said to be “split” if it is replicated such that each element referencing PID gets its own new DVAR. If a DVAR designing PID is not split, then that DVAR continues to design all of the elements referencing PID. By default, all DVARs designing PID are split. If this is not desired, then the NDVAR list can be used to specify design variables that should not be split, or alternatively, the DVAR list can be used to specify the only design variables that are to be split.
- The split design variables can be either independent or dependent. If the design variables are dependent then links between the design variables that design the same property are preserved in the generated DVARs. DLINK entries that attempt to links design variables from different topometry regions are not accepted and will generate an error message.
- Independent design variables that are split can be either continuous or discrete.
- Design variables that are split can be listed in DRESP2 data. The DVAR list on DRESP2 can contain split variables from only one topometry region. The list can contain regular (non-topometry) design variable and split design variables. When the program finds a split variable in a DRESP2 entry the program automatically generates new DRESP2 data for each internally created design variable.
- Design variables that are split can be listed in DRESP3 data pointing to a user-supplied subroutine. The DVAR list on such DRESP3 entries can contain split variables from only one topometry region. The list can contain regular (non-topometry) design variable and split design variables. When the program finds a split variable in a DRESP3 entry the program automatically generates new DRESP3 data for each internally created design variable.
- Design variables that are split can be listed in DRESP3 data pointing to built-in functions. The DVAR list on such DRESP3 entries can have design variables that belong to different topometry regions. The list can also contain regular design variables. If a split design variable is listed in this type of DRESP3, then all internally created design variables are added to the original list. In this case no additional DRESP3 data is generated.

15. Although it is possible, but is not recommended to list split design variables in DVGRID, DVGRIDC and/or DOMAIN data. This data remains unchanged. In other words no DVGRID, DVGRIDC or DOMAIN data are automatically generated by DSPLIT data. A warning message will be issued if split design variables are found on the DVGRID, DVGRIDC and/or DOMAIN data.
16. Properties referenced with DSPLIT data can also be referenced by **DTGRID** data. In other words, it is possible to perform topography and topometry optimization simultaneously on same elements.
17. DSPLIT data can be used in the same input data with any other shape, sizing and or topography design data.
18. To output the automatically generated data, use the solution control ECHO= SORT.
19. To output the updated model on the final, initial and/or intermediate design cycle use the solution control command **UPRINT**.
20. Symmetry conditions are not available for PK2UU and PM2UU regions.
21. If a COARSE continuation line with CTYPE = LENGTH is present, a SYM continuation line must also be used to define CID, even if no fabrication constraints are desired (all three TYPEi fields may be blank).

8.2.22 DTABLE

Data Entry: **DTABLE** - Table Constants.

Description: Define a table of constants that are frequently used in equations.

Format:

1	2	3	4	5	6	7	8	9	10
DTABLE	LABL1	VALU1	LABL2	VALU2	LABL3	VALU3	LABL4	VALU4	
+	LABL5	VALU5	LABL6	VALU6	-etc.-				

Example:

1	2	3	4	5	6	7	8	9	10
DTABLE	PI	3.1416	E	1.0E+6	RHO	0.1	NU	0.3	
+	BK	40.	ALPHA	0.001					

Field Information Description

2,4,..	LABLi	Unique identification label for the constant (non blank characters).
3,5,..	VALUi	Value of the constant (Real).

Remarks:

1. DTABLE data is referenced by **DVPROP2** and **DRESP2** to be used with **DEQATN** or by **DRESP3** to be used in the user-subroutine.
2. Multiple DTABLE entries may appear in the input data.
3. VALUi is referenced by using the corresponding LABLi for CIDI on the DVPROP2 data or NCi on the DRESP2 or DRESP3 data.
4. As an alternative to passing the tabled values through the argument list from DVPROP2 or DRESP2, the table constant name LABLi can be used directly in DEQATN data in the equation to represent the constant VALUi. In this case, the program will substitute VALUi for LABLi as long as LABLi is distinct from the argument names and function names.

8.2.23 DTGRID

Data Entry: **DTGRID** - Topography Data Entry.

Description: Define a region, methods and parameters to automatically generate design variables and perturbation vectors for shape optimization (topography optimization).

Format:

1	2	3	4	5	6	7	8	9	10
DTGRID	ID	LABEL	PTYPE	PID	EDGEM	PERTM	SOLM	RINIT	
+	"SHAPE"	STYPE	DIMEN1	DIMEN2	DIMEN3				
+	"SYM"	CID	TYPE1	TYPE2	TYPE3	n	SYMTOL		
+	"DVAR"	INIT	LB	UB	DELX	DXMIN			
+	"PERT"	GID	CID	COEFF	PX/G1	PY/G2	PZ/G3		
+	"DGRID"	DGID1	DGID2	DGID3	DGID4	DGID5	DGID6	DGID7	DGID8
+		DGID9	-etc.-						
+	"DGSET"	DSETID							
+	"NDGRID"	NGID1	NGID2	NGID3	NGID4	NGID5	NGID6	NGID7	NGID8
+		NGID9	-etc.-						
+	"NDGSET"	NSETID							
+	"BEADFR"	FRACT	BTYPE	TOL					

Example 1: Create perturbation on grids associated to all elements that reference PSHELL 100. The maximum height is 5.0

1	2	3	4	5	6	7	8	9	10
DTGRID	1012	LPanel	PSHELL	100					
+	SHAPE	CONE	5.0	3.0	4.0				

Example 2: Create perturbation on grids associated to all elements that reference PSHELL 100. The maximum height is 5.0. Use two mirror symmetries MZX and MYZ

1	2	3	4	5	6	7	8	9	10
DTGRID	1012	LPanel	PSHELL	100					
+	SHAPE	CONE	5.0	3.0	4.0				
+	SYM	1	MZX	MYZ	4.0				

Example 3: Create perturbation on grids associated to all elements that reference PCOMP 221. The maximum height is 4.0, the perturbations can only be on the positive direction of the surface (LB=0.0),

1	2	3	4	5	6	7	8	9	10
DTGRID	102	LPanel	PCOMP	221					
+	SHAPE	CONE	4.0	2.0	3.0				

DTGRID

Shape, Sizing, Topography, Topometry and Freeform Design Model Data

1	2	3	4	5	6	7	8	9	10
+	DVAR		0.0						

Example 4: Create perturbation on grids associated to all elements that reference PSKIN 555. The maximum height is 5.0. The perturbations can only be in the positive Z direction (LB=0.0, PX=0.0, PY=0.0, PZ=1.0)

1	2	3	4	5	6	7	8	9	10
DTGRID	103	LPanel	PSKIN	555					
+	SHAPE	CONE	5.0	2.0	1.0				
+	DVAR		0.0						
+	PERT				0.0	0.0	1.0		

Field Information Description

2	ID	Unique DTGRID region Identification number. (Integer > 0).
3	LABEL	User defined name for output purposes (Character or blank).
4	PTYPE	Designable region type. One of the following words: PROPSET, PSHELL, PCOMP, PSKIN or ESET.
5	PID	PROPSET, PSHELL, PCOMP, PSKIN or element SET identification number (Integer>0). Together with PTYPE, identifies the designable region. See Remark 3.
6	EDGEM	Word indicating desired boundary grid method: One of the following words: NOEDGE, INEDGE or EDGE. Default = NOEDGE. See Remark 4.
7	PERTM	Method used to calculate the perturbation and design variables generated by this topography region. One of the words LINK, BASIC, FFORM or blank. Default = LINK when DIMEN2 or DIMEN3 are nonzero, otherwise default = BASIC. See Remark 5.
8	SOLM	Method for calculating perturbation scale factor. Integer 0 or 2 or blank. Default=0. See Remark 6.
9	RINIT	Parameter to set design variable initial randomness (Real \geq 0.0 or -1.0 or blank. Default=-1.0). A value of 0.0 means that no initial randomness will be used. See remarks 10 and 11.
2	SHAPE	Key word indicating shape information. See Remark 7 and 21.
3	STYPE	Word indicating desired basic topography unit shape. One of the following words: CONE, LINE or UNIF.
4	DIMEN1	Maximum desired height (grid movement on the direction of the perturbation)(Real > 0.0).
5	DIMEN2	Minimum desired bead width (Real \geq 0.0 or blank. Default=0.0). This value is only used for STYPE=CONE.

5	DIMEN3	Bead pattern spread (Real ≥ 0.0 or blank. Default=0.0). This value only used for STYPE=CONe.
2	SYM	Word indicating that the following data correspond to symmetry topometry data. This data is optional. See Remark 8.
3	CID	Coordinate system identification number (Integer ≥ 0 or blank. Default = 0)
4	TYPE1	Fabrication constraint type. One of "MYZ", "MZX", "MXY", "CX", "CY", "CZ", "EX", "EY" or "EZ"., "UXY", "UYZ", "UZX" or "UXYZ", See Remarks 8 and 9
5,6	TYPE2, TYPE3	Fabrication constraint type. One of "MYZ", "MZX", "MXY", "CX", "CY", "CZ", "EX", "EY", "EZ", "UXY", "UYZ", "UZX", "UXYZ" or Blank. See Remarks 8 and 9.
7	n	Number of cyclic symmetry sections. (Integer ≥ 0 or Blank). If n=0 then axisymmetry is used. The value n is only used for "CX", "CY" or "CZ ". See remark 8.
8	SYMTOL	Tolerance for checking fabrication symmetry constraints in DTGRID. Default 0.001.
2	DVAR	Key word indicating design variable information type. See Remark 7.
3	INIT	Initial value or scale factor for initial value of internally generated design variables. (Real, Default=0.001 if RINIT = -1.0 or 0.0 otherwise). See Remark 10.
4	LB	Lower bound of internally generated design variable. (Real or blank. $LB \leq UB$, Default=-1.0). See Remark 12.
5	UB	Upper bound of internally generated design variable. (Real or blank. $UB \geq LB$, Default=1.0). See Remark 12.
6	DELX	Design variable fractional move limit (Real > 0.0 or blank, Default=0.0001).
7	DXMIN	Design variable minimum move limit factor (Real > 0.0 or blank, Default=0.20).
2	PERT	Key word indicating perturbation information. See Remarks 7 and 13.
3	GID	Grid number (Integer > 0). The grid should belong to the topography region. See Remark 13.
4	CID	Coordinate system identification number (Integer ≥ 0 or blank. Default = 0)
5	COEFF	Scale factor to accelerate (COEFF >1.0) or reduce (COEFF <1.0) the speed of convergence or the optimization process. This value will scale all perturbations in the topography region (Real, Default = 1.0). See remark 14.

DTGRID

Shape, Sizing, Topography, Topometry and Freeform Design Model Data

6-8	PX,PY,PZ	Perturbation vector for providing a unique direction for all the automatically generated perturbation vectors. PX, PY and PZ are in the in the CID coordinate system located at grid GID. Only the direction of this vector is used. (Real or blank, Default = 0.0).
6-8	G1,G2,G3	Grid identification numbers for alternative ways of providing a unique direction for all the automatically generated perturbation vectors (Integer > 0)
2	DGRID	Key word indicating that following are grids on the topography region that need to be designed. See Remarks 7 and 15.
3-10	DGIDi	Grid numbers (Integer > 0).
2	DGSET	Key word indicating that following is a set of grids on the topography region that need to be designed. See Remarks 7 and 15.
3	DSETID	Grid numbers (Integer > 0).
2	NDGRID	Key word indicating that following are grids on the topography region that need to be excluded from the design space. See Remarks 7 and 16.
3-10	NGIDi	Grid set identification (Integer > 0).
2	NDGSET	Key word indicating that following is a set of grids on the topography region that need to be exclude from the design space. See Remarks 7 and 16.
3	NSETID	Grid set identification (Integer > 0).
2	BEADFR	Key word indicating bead fraction constraint information. See Remarks 19 and 20.
3	FRACT	Bead fraction. (0.0<Real <=1.0)
4	BTYPE	One of the following types: LOWER, UPPER or BOTH.Upper is to get an upper bound on the bead fraction constraint, lower is to get a lower bound on the bead fraction constraint.Both is to get a lower and an upper bound on the bead fraction constraint. Default is UPPER.
5	TOL	Tolerance for satisfying bead fraction constraints. Default 0.005.

Remarks.

1. DTGRID data is used to automatically generate multiple perturbation vectors and their associated design variables. Unless the PERT word is used, the perturbation vectors are created so that their directions are parallel to the normals to the designable region. With the PERT option, all grid will have a unique direction provided by the user with the data defined in PERT.
2. Multiple DTGRID entries are allowed in the input data.
3. When PTYPE PROPSET is used, the boundary of the region is the global boundary of all elements referencing any property in the PROPSET. The boundary is used for the EDGEM options.
4. If EDGEM is NOEDGE, edge grids will not be designed. If EDGEM is INEDGE, edge grids will only be indirectly designed. If EDGEM is EDGE, edge grids will be designed.
5. If PERTM is LINK, dependent variables will be used. If PERTM is BASIC or FFORM, no dependent variables will be used. If PERTM is FFORM then STYPE cannot be UNIF. If PERTM is FFORM then DIMEN2 is used by the program to find all grid that will be designed with a given design variable. This is equivalent to the COARSE option in DSHAPE. The advantage of using FFORM versus BASIC or LINK is that FFORM does not require any extra constraints or dependent variables to enforce maximum height. Bead patterns created with FFORM can be more sharp than bead created by LINK or BASIC.
6. If SOLM=0, the scale factor is approximated. If SOLM=2, the Gauss-Seidel iterative solver is used to calculate the scale factor.
7. A continuation entry with the SHAPE keyword must be present. Continuation entries with other keywords (SYM, DVAR, PERT, DGRID, DGSET, NDGRID, NDGSET) are optional. The continuation entries may be in any order. Each continuation keyword may be used at most once in a single DTGRID entry.
8. Up to three fabrication constraints are allowed per DTGRID entry. Following is a table with available fabrication constraints.

TYPE	Description of Fabrication Constraints
MXZ	Mirror symmetry with respect to the XZ plane
MYZ	Mirror symmetry with respect to the YZ plane
MZX	Mirror symmetry with respect to the ZX plane
CX	Cyclic symmetry about the X axis (n>0) or Axisymmetry about the X axis (n=0)
CY	Cyclic symmetry about the Y axis (n>0) or Axisymmetry about the Y axis (n=0)

CZ	Cyclic symmetry about the Z axis ($n>0$) or Axisymmetry about the Z axis ($n=0$)
EX	Extrusion along the X axis
EY	Extrusion along the Y axis
EZ	Extrusion along the Z axis
UXY	Uniform in planes parallel to the XY plane
UYZ	Uniform in planes parallel to the YZ pane
UZX	Uniform in planes parallel to the ZX plane
UXYZ	Uniform in all directions

9. Not all fabrication constraints can be mixed together. Following is a table with the allowed combinations:

<p> MX+MYZ, MX+MZX, MYZ+MZX, MX+MYZ+MZX CX+MYZ, CY+MZX, CZ+MX </p>
<p> EX+MX, EX+MZX, EX+MX+MZX EY+MYZ, EY+MX, EY+MYZ+MX EZ+MZX, EZ+MYZ, EZ+MZX+MYZ </p>
<p> UYZ+MYZ, UZX+MZX, UXY+MX </p>

10. If RINIT is -1.0, then the INIT value is a scale factor to randomly generated initial design variable values. If RINIT is 0.0, then the INIT value is used as the initial design variable value. A positive RINIT value scales the range about the INIT value from which a random value is selected. Larger values of RINIT make it more likely that the initial value will be farther from the INIT value.
11. Random initial values are generated by a pseudo-random number generator that gives a repeatable, but system-dependent, sequence of values. Changing the value of the analysis parameter RANDOM will cause the generator to use a different sequence and will lead to different random initial values.
12. To force the grids to move only on the positive side of the surface use LB=0.0. To force the grids to move only on negative side of the surface use UB=0.0. The positive side of the surface is defined by the direction of the normals. The normals follows the right hand rule on grids of elements.
13. By default, the program uses the average of the element normals at each grid to compute the perturbation direction for that grid. If a continuation entry with the PERT keyword is used, then the perturbation direction for one reference grid is calculated and replicated to all grids in the designable region. There are four options for specifying the perturbation for the reference grid:
- Option 1: Average element normal at GID. If PX, PY, and PZ are all blank, then the normal direction at grid GID is calculated and used at all grids.

Option 2: User supplied perturbation. If PX, PY, and/or PZ are real, then they are taken as components of a vector in the CID coordinate system (default is basic) located at grid GID. The vector defines the perturbation direction for all grids.

Option 3: Direction between two grids. If G1 and G2 are integer and G3 is blank, then the perturbation direction is defined as the direction from G1 to G2.

Option 4: Normal to a plane. If G1, G2 and G3 are integer, then they are used to define a plane, and the perturbation direction is normal to that plane. The positive direction is defined with the right-hand rule in exactly the same way as on a CTRIA3 element.

14. The scale factor COEFF can also be used for the automatically generated perturbation directions. In this case, the only nonblank data provided with PERT should be COEFF.
15. Grids that are on the boundary of the region and are excluded from the design with the NOEDGE option of the EDGEM field can be included by listing them on the DGRID list or in a grid SET referenced by DGSET.
16. Grids on the topography region that need to be excluded from the design (e.g., grids where load and/or boundary conditions are applied) can be deselected by listing them on the NDGRID list or in a grid SET referenced by NDGSET.
17. To output the automatically generated data, use the **DVGRID= PRINT** command in solution control. This output contains the perturbation vectors using DVGRID format. The corresponding DVAR and DLINK data will also be printed.
18. DTGRID data can be used with any other design data. The perturbation vectors are calculated using the original location of the grids.
19. The BEADFR continuation data is optional. If BEADFR is not used, no constraints will be created.
20. The BEADFR (bead fraction) option usually produces more discrete bead patterns. The FRACT value, or bead fraction value, represents the fraction of the total number of grids in the region that move from the initial design. Using the BOTH option with FRACT=0.4 will produce an answer in which approximately 40% of the designable grids move to the maximum allowable height, while the other 60% do not move. Using the UPPER option with FRACT=0.4 will produce an answer in which the total grid movement is no more than 40% times the the maximum allowable height times the number of grids in the region. Using the LOWER option with FRACT=0.4 will produce an answer in which the total grid movement is at least 40% times the the maximum allowable height times the number of grids in the region. The UPPER option is recommended for most situations. The LOWER or BOTH options can be used to force the optimizer move the grids in cases where there are low sensitivity values (which can occur in flat structures) and where other options (increasing INIT value or COEFF) do not help.
21. If STYPE is UNIF a single design variable is created. All grids in the region will use identical perturbation vectors to create a uniform grid movements. In this case, symmetric fabrication constraints are ignored.

8.2.24 DVAR

Data Entry: **DVAR** - Design Variable.

Description: Define a design variable that may be modified during design optimization

Format:

1	2	3	4	5	6	7	8	9	10
DVAR	ID	LABEL	INIT	LB	UB	DELX	DXMIN		
+	DVSID	VTYPE	VVALUE						

Example:

1	2	3	4	5	6	7	8	9	10
DVAR	1	ROD1	5.0	0.01	100.0	0.5	0.05		

Field Information Description

2	ID	Unique design variable identification number (Integer > 0).
3	LABEL	User supplied name for printing purposes (or blank).
4	INIT	Initial value. (Real). See remarks 5 and 6.
5	LB	Lower bound. (Real. $LB \leq INIT$).
6	UB	Upper bound. (Real. $UB \geq INIT$).
7	DELX	Design variable fractional move limit (Real > 0 or blank). See remark 1.
8	DXMIN	Design variable minimum move limit factor (Real > 0 or blank). See remark 1.
2	DVSID	Discrete design variable set identification number.(Integer > 0 or "INT" or blank). See remarks 2 and 3.
3	VTYPE	Variability type for random variable. One of "VAR", "STDDEV" or "COV" or blank. Default is STDDEV when VVALUE is non blank. See Remarks 7 and 8.
4	VVALUE	Variability value for random variable (Real > 0 or blank). If VTYPE=VAR then value is variance. If VTYPE=STDDEV or blank then value is standard deviation. If VTYPE=COV then value is coefficient of variation

Remarks:

1. DELX and DXMIN default to 0.5 and 0.1 respectively, unless these values are specified on the DOPT entry (in which case they default to the values specified on DOPT). The minimum move limit values is defined as:

$$DXMIN = \begin{cases} DXMIN & \text{if } |DV_{\text{initial}}| \leq 1.0 \\ DXMIN|DV_{\text{initial}}| & \text{if } |DV_{\text{initial}}| > 1.0 \end{cases}$$

2. If DVSID is Integer>0, then the design variable will take only discrete values defined on the corresponding **DVSET** and **DVSET1** entries. If DVSID is INT, the design variable will take only integer values between the lower and upper bounds. If DVSID is blank, the continuation line may be omitted.
3. If DVSID is not blank, the design variable cannot be made dependent using a **DLINK** entry.
4. Discrete values outside the design variable bounds are ignored.
5. The initial value may be changed from INIT to LB or UB depending on the setting of the DOPT parameter DVINIT. The default is not to change the initial value.
6. If DVSID is not blank, and INIT is not a member of the associated discrete set, the initial value may be changed, depending on the setting of the DOPT parameter DVINIT2. If DVINIT2 is 0, the default, *GENESIS* will stop with a fatal error if INIT does not belong to the associated discrete set. If DVINIT2 is 1, *GENESIS* will use INIT even if it does not belong to the provided discrete set. If DVINIT2 is 2, *GENESIS* will replace INIT with the value from the discrete set closest to the provided value. If DVINIT2 is 3, *GENESIS* will replace the design variable initial value with the value from the discrete set closest to the provided value and then hold all discrete design variables as constants during the optimization. This option allows restarts of a problem that has both discrete and continuous variables to keep improving the design using only the continuous variables.
7. If one or more random variables exist in the input data, then reliability analysis will be automatically performed. Variables that are not made random, will be considered deterministic and will have a variance of 0.0. If no random variable exists then traditional deterministic variables will be used.
8. The variability types for random variables are defined as follows:

$$VAR = \text{Variance} = \text{Var}(X) = \frac{\sum_i^n (X_i - \mu_X)^2}{(n - 1)}$$

$$STDDEV = \text{Standard Deviation} = \sigma_X(X) = \sqrt{\text{Var}(X)}$$

$$COV = \text{Coefficient of Variance} = \text{COV}(X) = \frac{\sigma_X(X)}{\mu_X}$$

8.2.25 DVGRID

Data Entry: **DVGRID** - Design Variable to Grid Coordinate Relation.

Description: Define the relation between one design variable and coordinates of one grid point.

Format:

1	2	3	4	5	6	7	8	9	10
DVGRID	DVID	GID	CID	COEFF	N1	N2	N3	BASIS	

Example:

1	2	3	4	5	6	7	8	9	10
DVGRID	2	110	5	1.0	0.5	0.3	0.8		

Field Information Description

2	DVID	DVAR identification number (Integer > 0).
3	GID	Grid number (Integer > 0).
4	CID	Coordinate system identification number (Integer ≥ 0 or blank. Default = 0)
5	COEFF	Linear coefficients of relation between design variables and coordinates (Real, Default = 1.0. ignored if BASIS = 1).
6-8	N1,N2,N3	If BASIS = 0, components of a vector measured in the coordinate system defined by CID at the location of the grid defined by GID. If BASIS = 1, the location of the basis grid is in the CID coordinate system (Real or blank, Default = 0.0).
9	BASIS	For basis vectors, BASIS=1; For perturbation vectors, BASIS = 0. Default is DOPT parameter BASIS (0, 1 or blank).

Remarks:

1. See **Shape and Sizing Design Capabilities** (Ch. 3) for a discussion of Shape Optimization.
2. A CID of zero or blank references the basic coordinate system.
3. If BASIS=1, then the N1, N2, N3 define a basis vector and are measured from the origin of CID to the candidate grid point.
4. Only one DVGRID entry may reference a given grid for each design variable if BASIS=1.

5. The coordinate update equation when BASIS = 1, is given as

$$\begin{Bmatrix} x \\ y \\ z \end{Bmatrix} = \begin{Bmatrix} x_0 \\ y_0 \\ z_0 \end{Bmatrix} + \sum_I DVID_I \begin{Bmatrix} x_i - x_0 \\ y_i - y_0 \\ z_i - z_0 \end{Bmatrix}$$

where x_i, y_i, z_i are coordinates of N_1, N_2, N_3 in the basic coordinate system and x_0, y_0, z_0 are the grid coordinates of the original design in the basic coordinate system.

6. If BASIS = 0, then N1, N2, N3 define perturbations from the location of the grid point defined on the GRID data entry.
7. Multiple references to the same grid and design variable results in vectorial addition of participation coefficient vectors (BASIS = 0).
8. There is no restriction on the number of DVGRID data which may reference a given grid or design variable if BASIS = 0.
9. The CID, N1, N2, N3, define the direction of grid perturbation. The magnitude of the grid perturbation, with respect to unit value of design variable X_{DVID} , is the length of \vec{N} multiplied by COEFF, (BASIS = 0).
10. The coordinate update equation when BASIS = 0,:

$$\begin{Bmatrix} x \\ y \\ z \end{Bmatrix} = \begin{Bmatrix} x_0 \\ y_0 \\ z_0 \end{Bmatrix} + \sum_I DVID_I COEFF_i [T_0] \begin{Bmatrix} N_1 \\ N_2 \\ N_3 \end{Bmatrix}$$

where the direction cosines that define the $[T_0]$ matrix are determined from CID and the location of:

$$\begin{Bmatrix} x_0 \\ y_0 \\ z_0 \end{Bmatrix}$$

For example, if the CID defines a cylindrical coordinate system, the vector $N=(0, 2, 0)$ defines a vector of length 2 in the direction θ at the location of the grid point (X_0, Y_0, Z_0) , (N does not define a perturbation of 2 degrees).

8.2.26 DVGRIDC

Data Entry: **DVGRIDC** - Design Variable to Grid Coordinate Relationship Control.

Description: This entry contain a perturbation vector or basis vector to automatically generate DVGRID data for all DOMAIN elements that contains the DVAR and GRID IDs in their entry. The automatic generation will be based on isoparametric interpolation functions.

Format:

1	2	3	4	5	6	7	8	9	10
DVGRIDC	DVID	GID	CID	COEFF	N1	N2	N3	BASIS	

Example:

1	2	3	4	5	6	7	8	9	10
DVGRIDC	1	12	1	0.4	1.0	0.0	0.0		

Alternative Format:

1	2	3	4	5	6	7	8	9	10
DVGRIDC	DVID	GID		COEFF	G1	G2			

Example:

1	2	3	4	5	6	7	8	9	10
DVGRIDC	1	12		1.0	1001	1002			

Field	Information	Description
2	DVID	DVAR identification number (Integer > 0).
3	GID	Grid number (Integer > 0). The grid should belong to a DOMAIN element. See Remark 1.
4	CID	Coordinate system identification number (Integer ≥ 0 or blank. Default = 0)
5	COEFF	Linear coefficients of relation between design variables and coordinates (Real, Default = 1.0. ignored if BASIS = 0).
6-8	N1,N2,N3	If BASIS = 0, components of a vector measured in the coordinate system defined by CID at the location of the grid defined by GID. If BASIS = 1, the location of the basis grid is in the CID coordinate system (Real or blank, Default = 0.0).
6-7	G1,G2	Grid numbers (Integer > 0)
9	BASIS	For BASIS vectors BASIS=1; For perturbation vectors BASIS = 0. Default is DOPT parameter BASIS (0, 1 or blank).

Remarks:

1. The grid perturbed by this entry has to be either a corner grid of a DOMAIN element or an interior grid of a DOMAIN element that is close to the midpoint of an edge of the DOMAIN element. Otherwise, an error message will be issued and the program will stop.
2. Only one perturbation per edge of a DOMAIN element can be applied. Otherwise, the program will stop with an error message. In other words, having two perturbations applied to nodes close to the midpoint of one edge of a DOMAIN element is not allowed.
3. See **Basic Shape Design Capabilities** (Ch. 3) for a discussion of Shape Optimization.
4. A CID of zero or blank references the basic coordinate system.
5. If BASIS=1, then the N1, N2, N3 define a basis vector and are measured from the origin of CID to the grid point.
6. Only one DVGRIDC entry may reference a given grid for each design variable if BASIS=1.
7. The coordinate update equation when BASIS = 1, is given as

$$\begin{Bmatrix} x \\ y \\ z \end{Bmatrix} = \begin{Bmatrix} x_0 \\ y_0 \\ z_0 \end{Bmatrix} + \sum_I \text{DVID}_I \begin{Bmatrix} x_i - x_0 \\ y_i - y_0 \\ z_i - z_0 \end{Bmatrix}$$

where x_i, y_i, z_i are coordinates of N_1, N_2, N_3 in the basic coordinate system and x_0, y_0, z_0 are the grid coordinates of the original design in the basic coordinate system.

8. If BASIS = 0, then N1, N2, N3 define perturbations from the location of the grid point defined on the GRID data entry.
9. There is no restriction on the number of DVGRIDC data which may reference a given grid or design variable if BASIS = 0.
10. Multiple references to the same grid and design variable results in vectorial addition of participating coefficient vectors (BASIS = 0).
11. The CID, N1, N2, N3, define the direction of grid perturbation. The magnitude of the grid perturbation, with respect to unit value of design variable X_{DVID} , is the length of \vec{N} multiplied by COEFF, (BASIS = 0).
12. The alternative format defines a perturbation vector whose direction is from G1 to G2, and whose magnitude is COEFF.

13. The coordinate update equation is given as (BASIS = 0):

$$\begin{Bmatrix} x \\ y \\ z \end{Bmatrix} = \begin{Bmatrix} x_0 \\ y_0 \\ z_0 \end{Bmatrix} + \sum_i \text{DVID}_i \text{COEFF}_i [T_0] \begin{Bmatrix} N_1 \\ N_2 \\ N_3 \end{Bmatrix}$$

where the direction cosines that define the $[T_0]$ matrix are determined from CID and the location of:

$$\begin{Bmatrix} x_0 \\ y_0 \\ z_0 \end{Bmatrix}$$

For example, if the CID defines a cylindrical coordinate system, the vector $N=(0, 2, 0)$ defines a vector of length 2 in the direction θ at the location of the grid point (X_0, Y_0, Z_0) , (N does not define a perturbation of 2 degrees).

14. A grid is defined to be close to the midside node if its distance to the midpoint of an edge is less than 1/6 of the length of the edge. This definition can be changed using the optimization parameter DVGTOL. However, it is recommended to do the change with care to avoid numerical problems.

8.2.27 DVPROP1

Data Entry: **DVPROP1** - Design Variable to Property Relation.

Description: Define a linear relation between an analysis model property and design variables.

Format:

1	2	3	4	5	6	7	8	9	10
DVPROP1	ID	PID	FID	CO	PMIN	DELP	DPMIN		
+	DVID1	C1	DVID2	C2	DVID3	C3	DVID4	C4	
+	DVID5	C5	-etc.-						

Example:

1	2	3	4	5	6	7	8	9	10
DVPROP1	12	25	6	0.1	1.0	0.5	0.01		
+	4	1.0	6	10.0					

Field Information Description

2	ID	Unique identification number of the relation between a property and design variables. (Integer > 0).
3	PID	Property entry identification number. (Integer > 0).
4	FID	Position of the property in the element property table of the analysis model. (Integer > 0). See Remark 2.
5	CO	Constant term of relation. (Real or blank. Default = 0.0).
6	PMIN	Minimum value allowed for this property. (Real or blank. Default = 1.0E-10) If FID references a stress recovery location coefficient or I21, I31, I32 or a component of the offset vector of CONM3, I12 of a PBAR, material orientation of a PCOMP layer, or stiffness, conduction, mass or damping of PELAS, PVECTOR, PBUSH, PELASH, PMASS, PDAMP or PVISC then default = -1.0+35.
7	DELP	Property fractional move limit (Real > 0.0 or blank). (See Remark 3).
8	DPMIN	Property minimum move limit factor (Real > 0.0). (See Remark 3).
2,4,..	DVIDi	DVAR identification number. (Integer > 0). Only the first DVID is required. The rest can be blank.
3,5,..	Ci	Linear coefficient of relation. (Real or blank. Default = 1.0).

DVPROP1 Shape, Sizing, Topography, Topometry and Freeform Design Model Data

Remarks:

1. The relationship is given by: $\text{Property}_I = C_0 + \sum_i C_i DV_i$
2. FID identifies the position of an entry in the element property table. For example, to specify the area of PBAR or PBARL, FID=2. See tables below.
3. DELP and DPMIN default to 0.5 and 0.1 respectively, unless these values are specified in the DOPT entry (in which case they default to the values specified in DOPT). The minimum move limit value is defined as:

$$\text{Minimum Move Limit Value} = \begin{cases} \text{DPMIN} & \text{if } |P_{\text{initial}}| \leq 1.0 \\ \text{DPMIN}|P_{\text{initial}}| & \text{if } |P_{\text{initial}}| > 1.0 \end{cases}$$

4. It is strongly recommended that DVPROP1 data not be used to control plate/shell element properties (PSHELL). Changing the plate element thickness with DVPROP1 data will not automatically update the bending stiffness or stress recovery point locations. Two DVPROP1 data statements would be needed to update the stress recovery points and one DVPROP2 data statement with a DEQATN data statement would be needed to update the bending stiffness. It is recommended that the DVPROP3 data with TYPE = SOLID be used to design plate/shell structures.
5. The FID for composite layer thicknesses can be from 101 to 500. The FID for composite layer material orientation angles can be from 501 to 900.

Field Item Numbers for the FID Entry:

PCONM3 - Mass element properties.

FID	INFORMATION
1	X component of offset vector.
2	Y component of offset vector.
3	Z component of offset vector.
4	Mass value.
5	I11 value.
6	I21 value.
7	I22 value.
8	I31 value.
9	I32 value.
10	I33 value.

PDAMP - Scalar damping element properties.

FID	INFORMATION
1	The element damping coefficient.

PELAS - Elastic element properties.

FID	INFORMATION
1	The elastic element stiffness.
2	The elastic element stress recovery coefficient.
3	Structural damping coefficient

PELASH - Scalar conduction element properties.

FID	INFORMATION
1	The element conduction coefficient.

DVPROP1 Shape, Sizing, Topography, Topometry and Freeform Design Model Data

PVECTOR - CVECTOR element properties.

FID	INFORMATION
1	K_{11}
2	K_{12}
3	K_{13}
4	K_{14}
5	K_{15}
6	K_{16}
7	K_{22}
8	K_{23}
9	K_{24}
10	K_{25}
11	K_{26}
12	K_{33}
13	K_{34}
14	K_{35}
15	K_{36}
16	K_{44}
17	K_{45}
18	K_{46}
19	K_{55}
20	K_{56}
21	K_{66}
22	Structural Damping Coefficient, GE

PBUSH - CBUSH element properties.

FID	INFORMATION
1	Translational stiffness value in the element coordinate system, K_1
2	Translational stiffness value in the element coordinate system, K_2
3	Translational stiffness value in the element coordinate system, K_3
4	Rotational stiffness value in the element coordinate system, K_4
5	Rotational stiffness value in the element coordinate system, K_5
6	Rotational stiffness value in the element coordinate system, K_6
7	Translational viscous damping value in the element coordinate system, B_1
8	Translational viscous damping value in the element coordinate system, B_2
9	Translational viscous damping value in the element coordinate system, B_3
10	Rotational viscous damping value in the element coordinate system, B_4
11	Rotational viscous damping value in the element coordinate system, B_5
12	Rotational viscous damping value in the element coordinate system, B_6
13	Structural Damping Coefficient, GE
14	Stress recovery coefficient for translational components, ST
15	Stress recovery coefficient for rotation components, SR
15	Strain recovery coefficient for translational components, ET
17	Strain recovery coefficient for rotational components, ER

DVPROP1

Shape, Sizing, Topography, Topometry and Freeform Design Model Data

PHBDY - CHBDY element properties.

FID	INFORMATION
1	Area factor.
2	D1 diameter.
3	D2 diameter.
4	Absorptivity.

PMASS - Scalar mass element properties.

FID	INFORMATION
1	The element mass.

PROD - Rod element properties.

FID	INFORMATION
1	Rod element area.
2	Nonstructural mass per unit length.

PBAR - Bar element properties.

FID	INFORMATION
1	Nonstructural mass per unit length.
2	Bar element area.
3	Torsional constant.
4	I1 area moment of inertia (I_{zz}).
5	I2 area moment of inertia (I_{yy}).
6	I12 area moment of inertia.
7	AS1 shear area for plane 1 (AS_y).
8	AS2 shear area for plane 2 (AS_z).
9	C1 stress recovery coefficient.
10	C2 stress recovery coefficient.
11	D1 stress recovery coefficient.
12	D2 stress recovery coefficient.
13	E1 stress recovery coefficient.
14	E2 stress recovery coefficient.
15	F1 stress recovery coefficient.
16	F2 stress recovery coefficient.

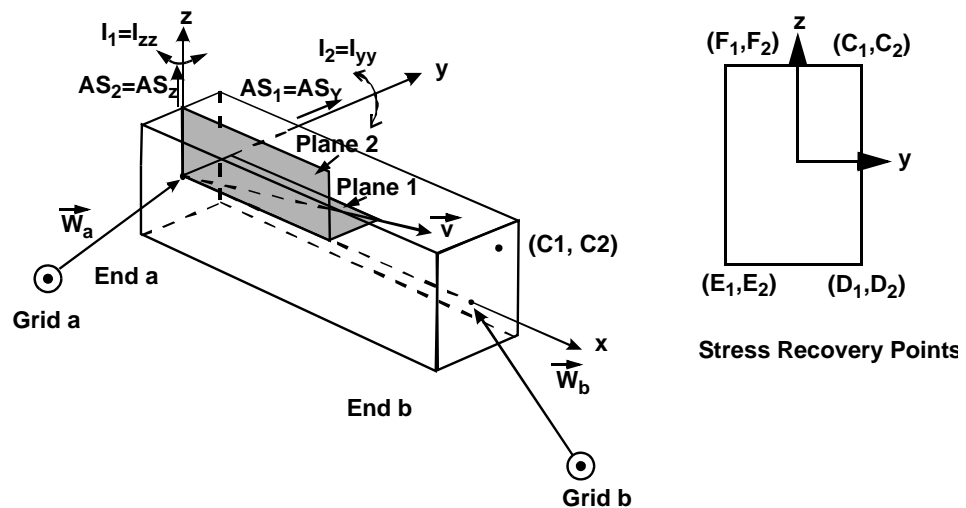


Figure 8-3

DVPROP1 Shape, Sizing, Topography, Topometry and Freeform Design Model Data

PSHELL - Shell element properties.

FID	INFORMATION
1	Nonstructural mass per unit area.
2	Plate/Shell element thickness.
3	Plate/Shell element bending stiffness.
4	Plate/shell element shear thickness.
5	Negative stress fiber distance (Z1).
6	Positive stress fiber distance (Z2).

PSHEAR - Shear Panel element properties.

FID	INFORMATION
1	Nonstructural mass per unit area.
2	Element thickness.
3	F1 extensional stiffness parameter
4	F2 extensional stiffness parameter

PVISC - Viscous damper element properties.

FID	INFORMATION
1	The extensional damping coefficient.
2	The rotational damping coefficient.

PCOMP/PCOMPG - Composite element properties.

FID	INFORMATION
3	Non-structural mass per unit area.
4	Location of the lower surface (Z_0).
7	Structural damping coefficient (GE).
100+i	Thickness of layer i.
500+i	Angle of layer i.

Shape, Sizing, Topography, Topometry and Freeform Design Model Data

PAXIS - Composite element properties.

FID	INFORMATION
1	Element property multiplier.

PK2UU - K2UU1 element properties.

FID	INFORMATION
1	Element property multiplier.

PM2UU - M2UU1 element properties.

FID	INFORMATION
1	Element property multiplier.

8.2.28 DVPROP2

Data Entry: **DVPROP2** - Design Variable to Property Relation.

Description: Define the relation between an analysis model property and design variables by means of a user supplied equation.

Format:

1	2	3	4	5	6	7	8	9	10
DVPROP2	ID	PID	FID	EQID	PMIN	DELP	DPMIN		
+	"DVAR"	DVID1	DVID2	DVID3	DVID4	DVID5	DVID6	DVID7	DVID8
+		DVID9	-etc.-						
+	"DTABLE"	NC1	NC2	NC3	NC4	NC5	NC6	NC7	NC8
+		NC9	-etc.-						

Example:

1	2	3	4	5	6	7	8	9	10
DVPROP2	13	120	5	12	0.1-5	0.4	0.01		
+	DVAR	4	6	12					
+	DTABLE	YOUNGS							

Field Information Description

2	ID	Unique identification number for the relation between a property and design variables. (Integer > 0).
3	PID	Property entry identification number. (Integer > 0).
4	FID	Position of the property in the element property table of the analysis model entry. (Integer > 0.)
5	EQID	DEQATN entry identification number. (Integer > 0).
6	PMIN	Minimum value allowed for this property. (Real. Default = 1.0E-10) If the property is a stress recovery location or coefficient of PBAR or PSHELL, the inertia terms I21, I31 or I32 of PCONM3, the product of inertia I12 of PBAR component of the offset vector of CONM3, or stiffness, conduction, mass, or damping of PELAS, PELASH, PVECTOR, PBUSH, PMASS, PDAMP or PVISC, the default = -1.0+35).
7	DELP	Property fractional move limit (Real > 0.0 or blank). See Remark 4.
8	DPMIN	Property minimum move limit factor (Real > 0.0 or blank). See Remark 4.
2	DVAR	Word indicating DVAR identification numbers.
3,4..	DVIDi	DVAR entry identification number. (Integer > 0).

2	DTABLE	Word indicating constant identification numbers in DTABLE. This field and the CIDs followed can be omitted if there are no constants involved in this relation.
3,4..	NCi	Name of constant defined in DTABLE input. (Character). See Remark 6.

Remarks:

1. The variables identified by DVIDi and CIDi correspond to parameters listed in the left-hand side of the equation on the DEQATN entry identified by EQID. The parameters are assumed to be in the order DVID1, DVID2,..., DVIDn, CID1, CID2,...CIDm.
2. “DVAR” must exist and must be placed before “DTABLE,” which is optional.
3. FID identifies the position of an entry in one of the tables below. For example, to specify the area of a PBAR, FID = 2. See tables below.
4. DELP and DPMIN default to 0.5 and 0.1 respectively, unless these values are specified on the DOPT entry (in which case they default to the values specified on DOPT). The minimum move limit value is defined as:

$$\text{Minimum Move Limit Value} = \begin{cases} \text{DPMIN} & \text{if } |P_{\text{initial}}| \leq 1.0 \\ \text{DPMIN}|P_{\text{initial}}| & \text{if } |P_{\text{initial}}| > 1.0 \end{cases}$$

5. The FID for composite layer thicknesses can be from 101 to 500. The FID for composite layer material orientation angles can be from 501 to 900.
6. The name CYCLE can be used in the DTABLE list to pass the current value of the design cycle number (as long as the name CYCLE is not explicitly defined on the DTABLE entry).

Field Item Numbers for the FID Entry:

PCONM3 - Mass element properties.

FID	INFORMATION
1	X component of offset vector.
2	Y component of offset vector.
3	Z component of offset vector.
4	Mass value.
5	I11 value.
6	I21 value.
7	I22 value.
8	I31 value.
9	I32 value.
10	I33 value.

PDAMP - Scalar damping element properties.

FID	INFORMATION
1	The element damping coefficient.

PELAS - Elastic element properties.

FID	INFORMATION
1	The elastic element stiffness.
2	The elastic element stress recovery coefficient.
3	Structural damping coefficient

PELASH - Scalar conduction element properties.

FID	INFORMATION
1	The element conduction coefficient.

PVECTOR - CVECTOR element properties.

FID	INFORMATION
1	K_{11}
2	K_{12}
3	K_{13}
4	K_{14}
5	K_{15}
6	K_{16}
7	K_{22}
8	K_{23}
9	K_{24}
10	K_{25}
11	K_{26}
12	K_{33}
13	K_{34}
14	K_{35}
15	K_{36}
16	K_{44}
17	K_{45}
18	K_{46}
19	K_{55}
20	K_{56}
21	K_{66}
22	Structural Damping Coefficient, GE

DVPROP2

Shape, Sizing, Topography, Topometry and Freeform Design Model Data

PBUSH - CBUSH element properties.

FID	INFORMATION
1	Translational stiffness value in the element coordinate system, K_1
2	Translational stiffness value in the element coordinate system, K_2
3	Translational stiffness value in the element coordinate system, K_3
4	Rotational stiffness value in the element coordinate system, K_4
5	Rotational stiffness value in the element coordinate system, K_5
6	Rotational stiffness value in the element coordinate system, K_6
7	Translational viscous damping value in the element coordinate system, B_1
8	Translational viscous damping value in the element coordinate system, B_2
9	Translational viscous damping value in the element coordinate system, B_3
10	Rotational viscous damping value in the element coordinate system, B_4
11	Rotational viscous damping value in the element coordinate system, B_5
12	Rotational viscous damping value in the element coordinate system, B_6
13	Structural Damping Coefficient, GE
14	Stress recovery coefficient for translational components, ST
15	Stress recovery coefficient for rotation components, SR
15	Strain recovery coefficient for translational components, ET
17	Strain recovery coefficient for rotational components, ER

PHBDY - CHBDY element properties.

FID	INFORMATION
1	Area factor.
2	D1 diameter.
3	D2 diameter.
4	Absorptivity.

PMASS - Scalar mass element properties.

FID	INFORMATION
1	The element mass.

PROD - Rod element properties.

FID	INFORMATION
1	Rod element area.
2	Nonstructural mass per unit length.

PBAR - Bar element properties.

FID	INFORMATION
1	Nonstructural mass per unit length.
2	Bar element area.
3	Torsional constant.
4	I1 area moment of inertia (Izz).
5	I2 area moment of inertia (Iyy).
6	I12 area moment of inertia.
7	AS1 shear area for plane 1 (ASy).
8	AS2 shear area for plane 2 (ASz).
9	C1 stress recovery coefficient.
10	C2 stress recovery coefficient.
11	D1 stress recovery coefficient.
12	D2 stress recovery coefficient.
13	E1 stress recovery coefficient.
14	E2 stress recovery coefficient.
15	F1 stress recovery coefficient.
16	F2 stress recovery coefficient.

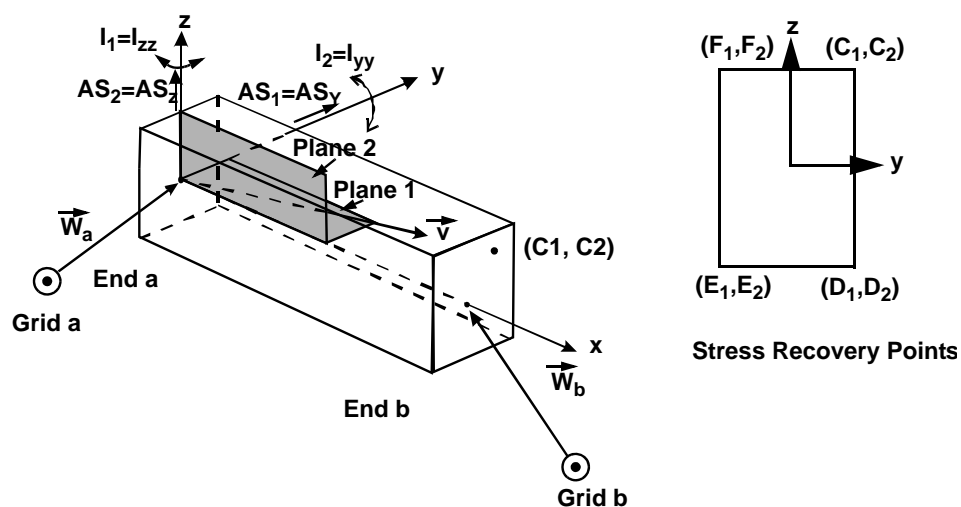


Figure 8-4

PSHELL - Shell element properties.

FID	INFORMATION
1	Nonstructural mass per unit area.
2	Plate/Shell element thickness.
3	Plate/Shell element bending stiffness.
4	Plate/shell element shear thickness.
5	Negative stress fiber distance.
6	Positive stress fiber distance.

PSHEAR - Shear Panel element properties.

FID	INFORMATION
1	Nonstructural mass per unit area.
2	Element thickness.
3	F1 extensional stiffness parameter
4	F2 extensional stiffness parameter

PVISC - Viscous damper element properties.

FID	INFORMATION
1	The extensional damping coefficient.
2	The rotational damping coefficient.

PCOMP/PCOMPG - Composite element properties.

FID	INFORMATION
3	Non-structural mass per unit area.
4	Location of the lower surface (Z_0).
7	Structural damping coefficient (GE).
100+i	Thickness of layer i.
500+i	Angle of layer i.

DVPROP2

Shape, Sizing, Topography, Topometry and Freeform Design Model Data

PAXIS - Axi-Symmetric element properties.

FID	INFORMATION
1	Element property multiplier.

PK2UU - K2UU1 element properties.

FID	INFORMATION
1	Element property multiplier.

PM2UU - M2UU1 element properties.

FID	INFORMATION
1	Element property multiplier.

8.2.29 DVPROP3

Data Entry: **DVPROP3** - Design Variable to Property Relation.

Description: Define the relation between a bar or plate analysis model property and design variables by means of element cross sections contained in the design model library.

Format:

1	2	3	4	5	6	7	8	9	10
DVPROP3	ID	PID	ELTYPE	ISHEAR	PMIN	DELP	DPMIN		
+	CSD1	CSD2	CSD3	CSD4	CSD5	CSD6	CSD7	CSD8	CSD9
+	CSD10	-etc.-							

Example:

1	2	3	4	5	6	7	8	9	10
DVPROP3	14	812	IBEAM		0.1	0.5	0.01		
+	4	1.0	2.5	1.0					

Field Information Description

2	ID	Unique identification number for the relation between a property (PBAR, PBARL or PSHELL) and design variables (Integer > 0).
3	PID	Property data identification number (Integer > 0).
4	ELTYPE	Bar or plate cross-section type contained in the element library (Only the following are allowed in this field: SQUARE, RECT, CIRCLE, TUBE, SPAR, BOX3, BOX4, IBEAM, RAIL, TEE, ANGLE, and for plates SOLID, SAND and SAND2, or Integer. For $1 \leq \text{ELTYPE} \leq 14$, see tables after remarks. For user defined elements, $15 \leq \text{ELTYPE} \leq 25$). See remark 5.
5	ISHEAR	Shear deformation switch. ISHEAR=1 to include shear deformation (0 or 1 or blank. Default = 0). See Remark 6.
6	PMIN	Minimum value allowed for the positive member properties during approximate optimization. (Real > 0. Default = 1.0E-10).
7	DELP	Property fractional move limit. Ignored for stress recovery locations and I_{yz} . (Real > 0.0 or blank). See Remark 2.
8	DPMIN	Property minimum move limit (Real > 0.0 or blank). See Remark 2.
2,3,4,...	CSDi	DVAR data identification number (Integer > 0) or, if constant, value of physical sizing variable (Real > 0.0). See remark 1.

DVPROP3 Shape, Sizing, Topography, Topometry and Freeform Design Model Data

Remarks:

1. If CSDi is an integer, it is assumed to be a design variable ID. If CSDi is a real value, then the cross sectional dimension has that constant value and is not changed during optimization. At least one CSDi must reference a design variable.
2. DELP and DPMIN default to 0.5 and 0.1 respectively, unless these values are specified on the DOPT entry (in which case they default to the values specified on DOPT). The minimum move limit value is defined as:

$$DPMIN = \begin{cases} DPMIN & \text{if } |P_{\text{initial}}| \leq 1.0 \\ DPMIN|P_{\text{initial}}| & \text{if } |P_{\text{initial}}| > 1.0 \end{cases}$$

3. The CSDi parameters are taken in order as shown on the figure for the particular beam element. For example, if ELTYPE=RECT, CSD1 is the width B, and CSD2 is the height, H.
4. A description of the beam elements and their cross sectional dimensions is given below. In the cross-section figures, the design variables are chosen to insure that the optimization will produce a buildable section. This requires that we treat “internal” dimensions as design variables. If the total height (for example) of a box beam (h + 2t) must not exceed a specified value H_{max} , this can be done by using a DRESP2 statement to create h+2t and a DCONS statement to limit the overall dimension to the specified value.
5. If a user defined element is used, then there must be a **DLIB** data entry for each different ELTYPE.
6. If ISHEAR=0 for BAR elements, the shear areas will be ignored. If ISHEAR=1 for BAR elements, the I12 area moment will be set to 0.0. If ISHEAR=0 for plate/shell elements, the transverse shear material ID (MID3) must be blank on the PSHELL data. If ISHEAR=1 for plate/shell elements, the transverse shear material ID (MID3) must be nonblank on the PSHELL data.
7. If the I12 field is nonzero on a PBAR data entry referenced by a DVPROP3 BAR element that has I12=0, the I12 area moment will be set to 0.0.
8. User defined elements are described in Chapter 3 **Element Library Design Variable to Property Relationships (DVPROP3)** (p. 105). There it is explained how to use the GNLBii routines to define cross sections as functions of design variables and how to use the GN40ii routines (p. 292) to define stresses as a function of cross sections and forces.

BAR ELEMENTS		
ELEMENT ID	ELEMENT TYPE	DESIGN VARIABLES
1	SQUARE	B
2	RECTANGLE	B, H
3	CIRCLE	D
4	TUBE	Di, t
5	SPAR	Ac, H, t
6	BOX3	b, t, h
7	BOX4	b, t1, h, t2
8	IBEAM	b, t1, h, t2
9	RAIL	b1, t1, b2, t2, h, t3
10	TEE	b, t1, h, t2
11	ANGLE	b, t1, h, t2

PLATE ELEMENTS		
ELEMENT ID	ELEMENT TYPE	DESIGN VARIABLES
12	SOLID	t
13	SAND	t, h
14	SAND2	t1, t2, h

The basic bar element orientation and coordinate system is shown below. In the figures defining the elements, the stress calculation points are shown on the positive face. That is, the stresses defined here are on the positive X face of the element at end b. At end a, the same definitions apply by considering the positive face at $X=0+$ (to visualize these stress locations at end A, look at the positive face of the element at a very small value of X).

The stresses are defined as positive if their directions are coincident with the forces that produce them. E.g. longitudinal stress is positive if F_x is positive (tension stress is positive).

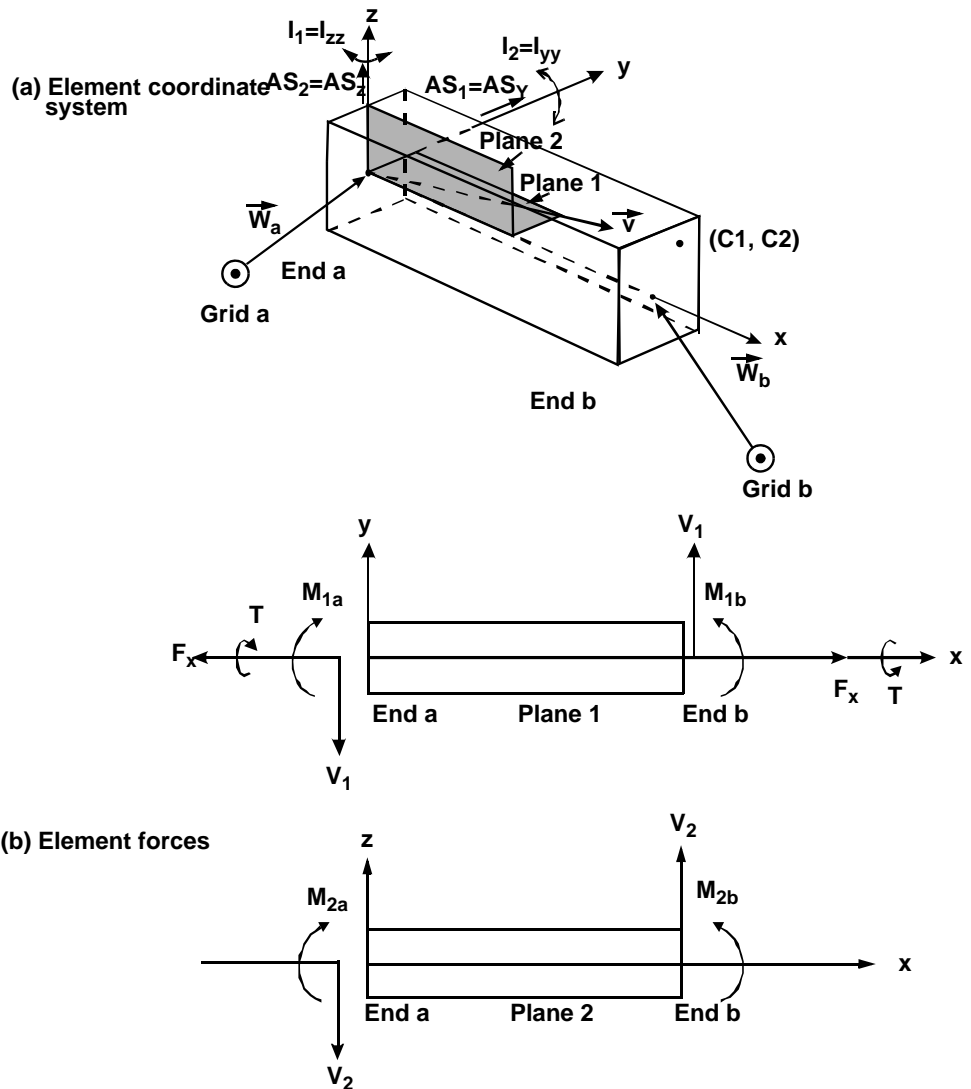


Figure 8-5 Geometry and Forces

Type 1: Square

Description: This is a solid, square element defined by a single design variable, B. Eight stresses are calculated at each end of the element. These include four bending stresses and four shear stresses.

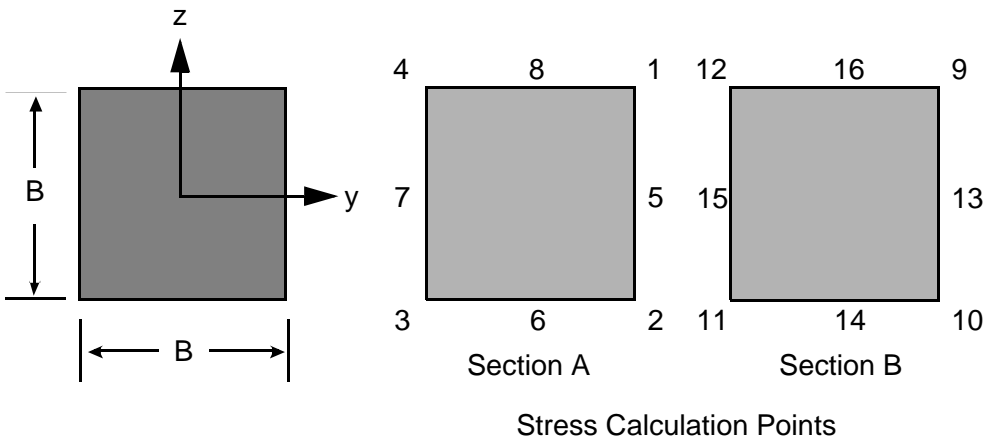


Figure 8-6

Design Variables:

Design Variable	Dimension
1	B

Section Properties:

$$A = B^2$$

$$I_{yy} = \frac{A^2}{12}$$

$$I_{zz} = \frac{A^2}{12}$$

$$J = 0.1406A^2$$

$$AS_y = \frac{10(1 + \nu)A}{12 + 11\nu}$$

$$AS_z = AS_y$$

DVPROP3

Shape, Sizing, Topography, Topometry and Freeform Design Model Data

Stress Calculations:

Location	Component	Value
1, 9	σ_x	$-\frac{M_z B}{2I_{zz}} - \frac{M_y B}{2I_{yy}} + \frac{F_x}{A}$
2, 10	σ_x	$-\frac{M_z B}{2I_{zz}} + \frac{M_y B}{2I_{yy}} + \frac{F_x}{A}$
3, 11	σ_x	$\frac{M_z B}{2I_{zz}} + \frac{M_y B}{2I_{yy}} + \frac{F_x}{A}$
4, 12	σ_x	$\frac{M_z B}{2I_{zz}} - \frac{M_y B}{2I_{yy}} + \frac{F_x}{A}$
5, 13	τ_{xz}	$\frac{0.676M_x B}{J} + \frac{3F_z}{2A}$
6, 14	τ_{xy}	$\frac{0.676M_x B}{J} + \frac{3F_y}{2A}$
7, 15	τ_{xz}	$-\frac{0.676M_x B}{J} + \frac{3F_z}{2A}$
8, 16	τ_{xy}	$-\frac{0.676M_x B}{J} + \frac{3F_y}{2A}$

Type 2: Rectangle

Description: This is a solid, rectangular element defined by design variables, B and H. Eight stresses are calculated at each end of the element. These include four bending stresses and four shear stresses.

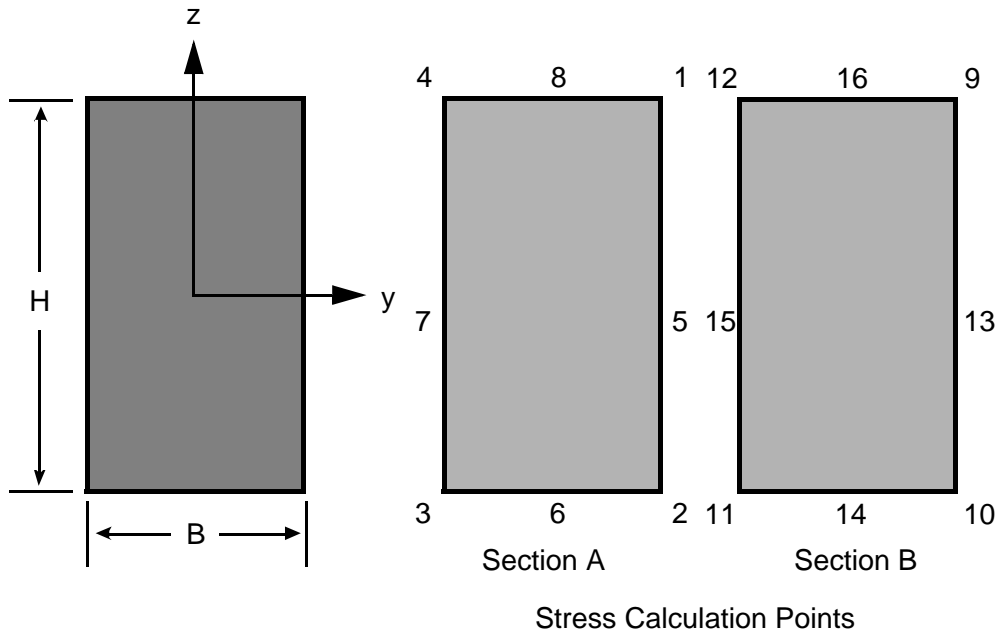


Figure 8-7

Design Variables:

Design Variable	Dimension
1	B
2	H

DVPROP3

Shape, Sizing, Topography, Topometry and Freeform Design Model Data

Section Properties:

$$A = BH$$

$$I_{yy} = \frac{BH^3}{12}$$

$$I_{zz} = \frac{HB^3}{12}$$

$$J = \left[0.1406 + 0.1148462 \text{TAN}^{-1} \left(\frac{B}{H} - 1 \right) \right] BH^3 \quad \text{if } B \geq H$$

$$J = \left[0.1406 + 0.1148462 \text{TAN}^{-1} \left(\frac{H}{B} - 1 \right) \right] HB^3 \quad \text{if } B < H$$

$$AS_y = \frac{10(1 + \nu)A}{12 + 11\nu}$$

$$AS_z = AS_y$$

Stress Calculations:

Location	Component	Value
1, 9	σ_x	$-\frac{M_z B}{2I_{zz}} - \frac{M_y H}{2I_{yy}} + \frac{F_x}{A}$
2, 10	σ_x	$-\frac{M_z B}{2I_{zz}} + \frac{M_y H}{2I_{yy}} + \frac{F_x}{A}$
3, 11	σ_x	$\frac{M_z B}{2I_{zz}} + \frac{M_y H}{2I_{yy}} + \frac{F_x}{A}$
4, 12	σ_x	$\frac{M_z B}{2I_{zz}} - \frac{M_y H}{2I_{yy}} + \frac{F_x}{A}$
5, 13	τ_{xz}	$\frac{k_1 M_x B}{J} + \frac{3F_z}{2A}$
6, 14	τ_{xy}	$\frac{k_2 M_x H}{J} + \frac{3F_y}{2A}$
7, 15	τ_{xz}	$-\frac{k_1 M_x B}{J} + \frac{3F_z}{2A}$
8, 16	τ_{xy}	$-\frac{k_2 M_x H}{J} + \frac{3F_y}{2A}$

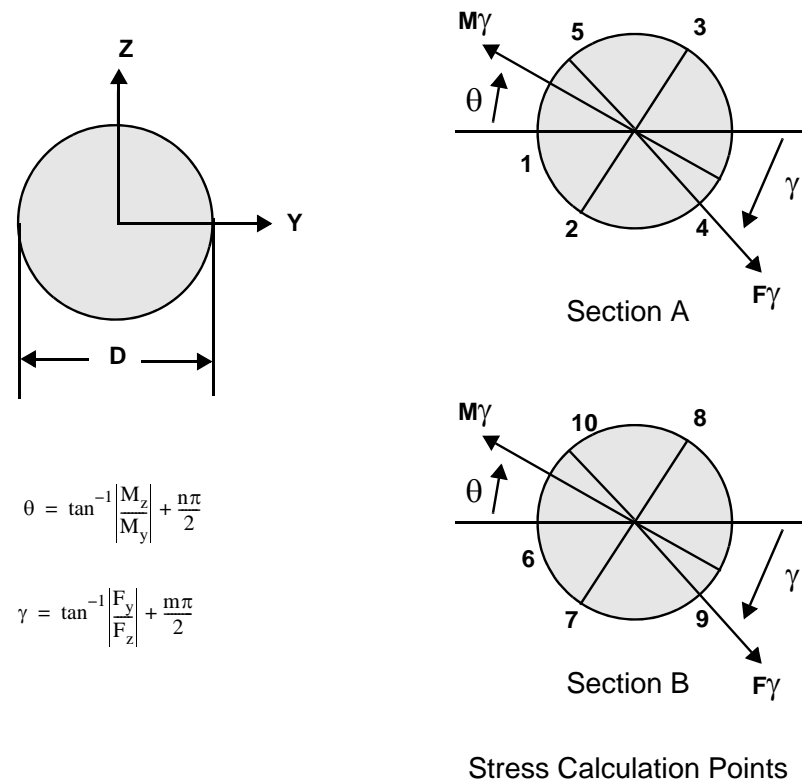
where

$$k_1 = 1.0 - \frac{8}{\pi^2 \cosh(\pi H/2B)}$$

$$k_2 = 1.0 - \frac{8}{\pi^2 \cosh(\pi B/2H)}$$

Type 3: Circle

Description: This is a solid, round element defined by a single design variable, D. Five stresses are calculated at each end of the element. These include von Mises (location 1), two bending stresses (locations 2-3) and two shear stresses (locations 4,5).



	$M_z > 0$	$M_z > 0$	$M_z < 0$	$M_z < 0$
	$M_y > 0$	$M_y < 0$	$M_y < 0$	$M_y > 0$
n	0	1	2	3

	$F_y > 0$	$F_y > 0$	$F_y < 0$	$F_y < 0$
	$F_z > 0$	$F_z < 0$	$F_z < 0$	$F_z > 0$
m	0	1	2	3

Figure 8-8

Design Variables:

Design Variable	Dimension
1	D

Section Properties:

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

$$I_{zz} = I_{yy} = \frac{\pi D^4}{64}$$

$$J = 2I_z$$

$$AS_y = \frac{6(1+\nu)A}{7+6\nu}$$

$$AS_z = AS_y$$

Stress Calculations:

Location	Component	Value
1, 6	σ_{vm}	$\max_{\alpha} \sqrt{\sigma_x^2 + 3\tau_{x\alpha}^2}$ $\sigma_x = \frac{D\sqrt{M_y^2 + M_z^2}}{2I_{yy}} \sin(\alpha - \theta) + \frac{F_x}{A}$ $\tau_{x\alpha} = \frac{M_x D}{2J} - \frac{4\sqrt{F_y^2 + F_z^2}}{3A} \sin(\alpha - \gamma)$
2, 7	σ_x	$\frac{D\sqrt{M_y^2 + M_z^2}}{2I_{yy}} + \frac{F_x}{A}$
3, 8	σ_x	$-\frac{D\sqrt{M_y^2 + M_z^2}}{2I_{yy}} + \frac{F_x}{A}$
4, 9	$\tau_{xz'}$	$\frac{M_x D}{2J} + \frac{4\sqrt{F_y^2 + F_z^2}}{3A}$
5, 10	$\tau_{xz'}$	$-\frac{M_x D}{2J} + \frac{4\sqrt{F_y^2 + F_z^2}}{3A}$

Type 4: Tube

Description: This is a tube (thick or thin) defined by two design variables, D_i and t . Five stresses are calculated at each end of the element, at the maximum bending and shear stress locations. These include three bending stresses (locations 1-3) and two shear stresses (locations 4,5).

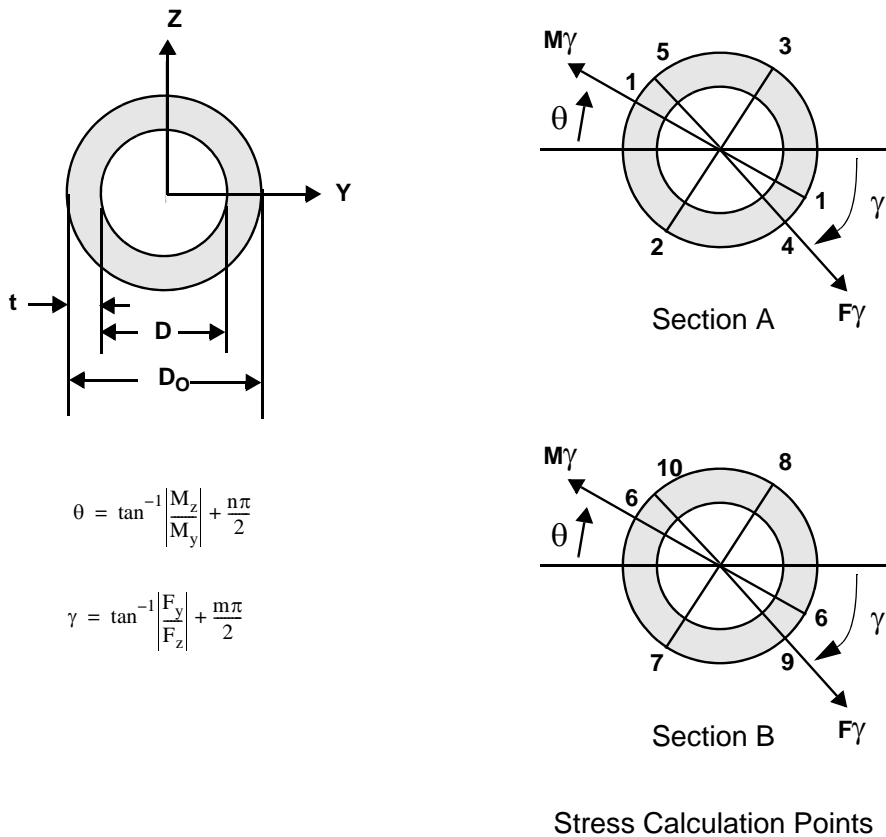


Figure 8-9

Design Variables:

Design Variable	Dimension
1	D_i
2	t

Other Dimensions:

$$D_o = D_i + 2t$$

Section Properties:

$$A = \pi(D_i + t)t$$

$$I_{zz} = I_{yy} = \frac{\pi(D_i^3 t + 3D_i^2 t^2 + 4D_i t^3 + 2t^4)}{8}$$

$$J = 2I_{zz}$$

$$AS_y = \frac{6(1 + \nu)(1 + m^2)^2 A}{(7 + 6\nu)(1 + m^2)^2 + (20 + 12\nu)m^2}$$

$$AS_z = AS_y$$

where

$$m = \frac{D_i}{D_i + 2t}$$

Stress Calculations:

Location	Component	Value
1, 6	σ_x	$\frac{F_x}{A}$
2, 7	σ_x	$\frac{R_0 \sqrt{M_y^2 + M_z^2}}{I_{yy}} + \frac{F_x}{A}$
3, 8	σ_x	$-\frac{R_0 \sqrt{M_y^2 + M_z^2}}{I_{yy}} + \frac{F_x}{A}$
4, 9	$\tau_{xz'}$	$\frac{M_x R_0}{J} + \frac{4(R_0^2 + R_0 R_i + R_i^2) \sqrt{F_y^2 + F_z^2}}{3(R_0^2 + R_i^2)A}$
5, 10	$\tau_{xy'}$	$-\frac{M_x R_0}{J} + \frac{4(R_0^2 + R_0 R_i + R_i^2) \sqrt{F_y^2 + F_z^2}}{3(R_0^2 + R_i^2)A}$

where $R_i = \frac{D_i}{2}$ $R_o = \frac{D_i}{2} + t$

Type 5: Spar

Description: This element is a frame element with the cross section described by the sizing variables A_c , H and t . It is intended primarily for use in planar frame structures. Three stresses are computed at each end of the element. These include two bending stresses and one shear stress.

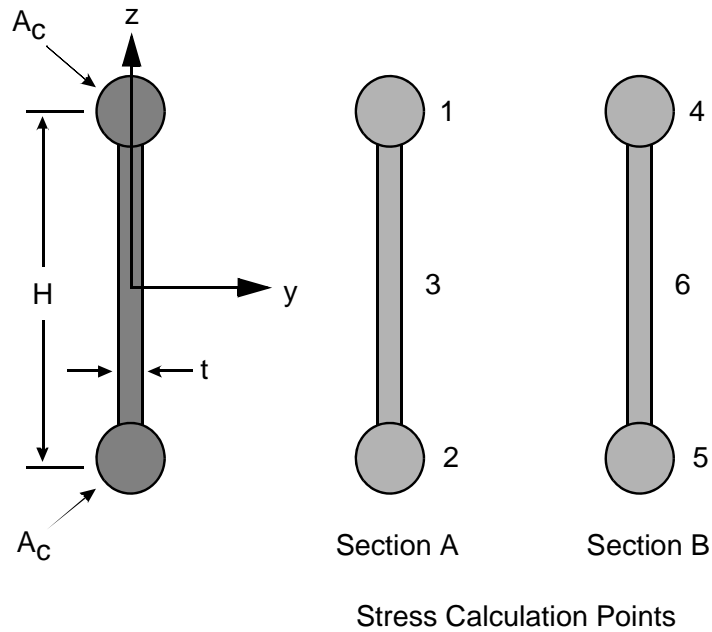


Figure 8-10

Design Variables:

Design Variable	Dimension
1	A_c
2	H
3	t

Section Properties:

$$A = tH + 2A_c$$

$$I_{yy} = \frac{tH^3}{12} + \frac{A_c H^2}{2} + 0.1591549 A_c^2$$

$$I_{zz} = \frac{Ht^3}{12} + 0.1591549 A_c^2$$

$$J = \frac{Ht^3}{3}$$

$$AS_y = 0.86A$$

$$AS_z = \frac{10(1 + \nu)(1 + 3m)^2 A}{12 + 72m + 150m^2 + 90m^3 + \nu(11 + 66m + 135m^2 + 90m^3)}$$

where

$$m = \frac{2A_c}{Ht}$$

Stress Calculations:

Location	Component	Value
1, 4	σ_x	$-\frac{M_y H}{2I_{yy}} + \frac{F_x}{A}$
2, 5	σ_x	$\frac{M_y H}{2I_{yy}} + \frac{F_x}{A}$
3, 6	$\tau_{xz'}$	$\frac{3F_z}{2tH}$

Type 6: Box3 (Constant Thickness Box)

Description: This element is a frame element with the cross section described by the sizing variables b , h and t . Eight stresses are computed at each end of the element. These include four bending stresses and four shear stresses.

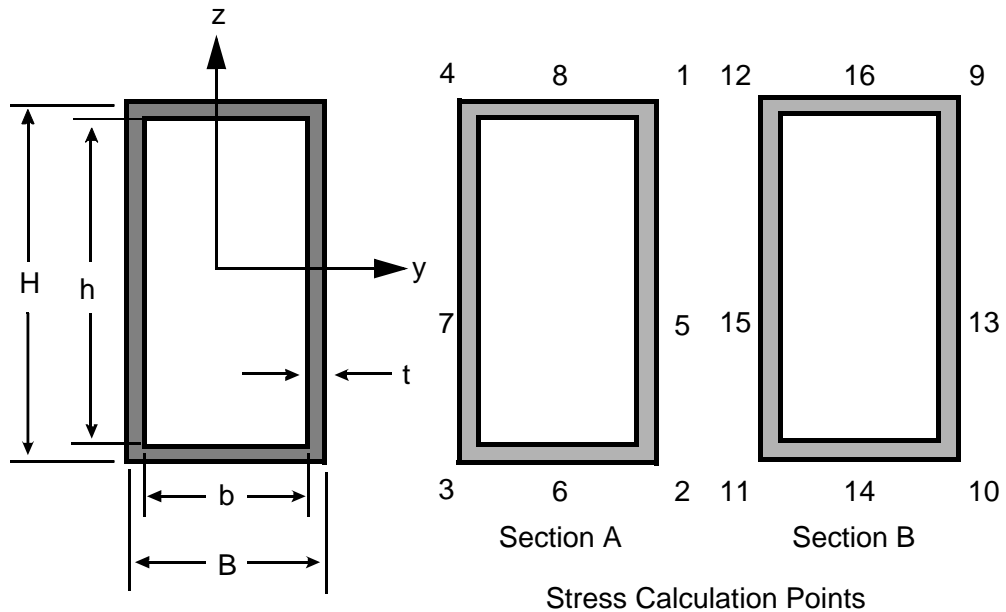


Figure 8-11

Design Variables:

Design Variable	Dimension
1	b
2	t
3	h

Other Dimensions:

$$H = h + 2t$$

$$B = b + 2t$$

Section Properties:

$$A = 2t(H + B - 2t)$$

$$I_{yy} = \frac{BH^3}{12} - \frac{(B - 2t)(H - 2t)^3}{12}$$

$$I_{zz} = \frac{HB^3}{12} - \frac{(H - 2t)(B - 2t)^3}{12}$$

$$J = \frac{2t(B - t)^2(H - t)^2}{B + H - 2t}$$

$$AS_y = \frac{20(1 + \nu)Bt}{12 + 11\nu}$$

$$AS_z = \frac{20(1 + \nu)Ht}{12 + 11\nu}$$

Stress Calculations:

Location	Component	Value
1, 9	σ_x	$-\frac{M_z B}{2I_{zz}} - \frac{M_y H}{2I_{yy}} + \frac{F_x}{A}$
2, 10	σ_x	$-\frac{M_z B}{2I_{zz}} + \frac{M_y H}{2I_{yy}} + \frac{F_x}{A}$
3, 11	σ_x	$\frac{M_z B}{2I_{zz}} + \frac{M_y H}{2I_{yy}} + \frac{F_x}{A}$
4, 12	σ_x	$\frac{M_z B}{2I_{zz}} - \frac{M_y H}{2I_{yy}} + \frac{F_x}{A}$
5, 13	τ_{xz}	$\frac{M_x}{2t(B-t)(H-t)} + \frac{3F_z}{4tH}$
6, 14	τ_{xy}	$\frac{M_x}{2t(B-t)(H-t)} + \frac{3F_y}{4tB}$
7, 15	τ_{xz}	$\frac{-M_x}{2t(B-t)(H-t)} + \frac{3F_z}{4tH}$
8, 16	τ_{xy}	$\frac{-M_x}{2t(B-t)(H-t)} + \frac{3F_y}{4tB}$

Type 7: Box4 (Two Thickness Box)

Description: This element is a frame element with the cross section described by the sizing variables, b , h , t_1 and t_2 . Eight stresses are computed at each end of the element. These include four bending stresses and four shear stresses.

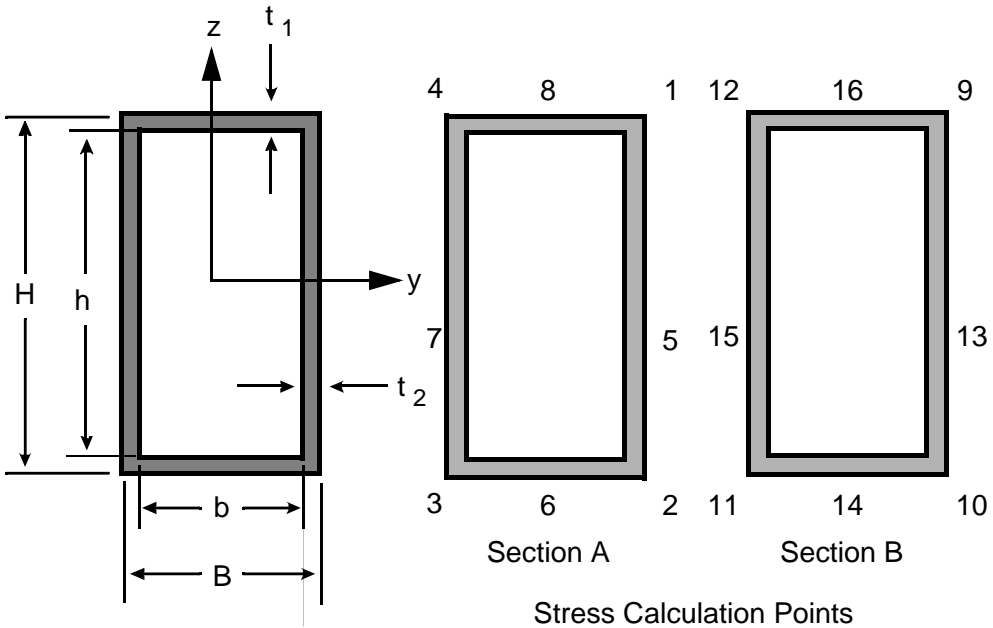


Figure 8-12

Design Variables:

Design Variable	Dimension
1	b
2	t_1
3	h
4	t_2

Other Dimensions:

$$H = h + 2t_1$$

$$B = b + 2t_2$$

Section Properties:

$$A = BH - (H - 2t_1)(B - 2t_2)$$

$$I_{yy} = \frac{BH^3 - (B - 2t_2)(H - 2t_1)^3}{12}$$

$$I_{zz} = \frac{HB^3 - (H - 2t_1)(B - 2t_2)^3}{12}$$

$$J = \frac{2t_1t_2(B - t_2)^2(H - t_1)^2}{t_1H + t_2B - t_1^2 - t_2^2}$$

$$AS_y = \frac{20(1 + \nu)Bt_1}{12 + 11\nu}$$

$$AS_z = \frac{20(1 + \nu)Ht_2}{12 + 11\nu}$$

Stress Calculations:

Location	Component	Value
1, 9	σ_x	$-\frac{M_z B}{2I_{zz}} - \frac{M_y H}{2I_{yy}} + \frac{F_x}{A}$
2, 10	σ_x	$-\frac{M_z B}{2I_{zz}} + \frac{M_y H}{2I_{yy}} + \frac{F_x}{A}$
3, 11	σ_x	$\frac{M_z B}{2I_{zz}} + \frac{M_y H}{2I_{yy}} + \frac{F_x}{A}$
4, 12	σ_x	$\frac{M_z B}{2I_{zz}} - \frac{M_y H}{2I_{yy}} + \frac{F_x}{A}$
5, 13	τ_{xz}	$\frac{M_x}{2t_2(B - t_2)(H - t_1)} + \frac{3F_z}{4t_2 H}$
6, 14	τ_{xy}	$\frac{M_x}{2t_1(B - t_2)(H - t_1)} + \frac{3F_y}{4t_1 B}$
7, 15	τ_{xz}	$\frac{-M_x}{2t_2(B - t_2)(H - t_1)} + \frac{3F_z}{4t_2 H}$
8, 16	τ_{xy}	$\frac{-M_x}{2t_1(B - t_2)(H - t_1)} + \frac{3F_y}{4t_1 B}$

Type 8: I Beam

Description: This element is a frame element with the cross section described by the sizing variables B , t_1 , h and t_2 . It is intended primarily for use in planar frame structures. Seven stresses are computed at each end of the element. These include four bending stresses and three shear stresses.

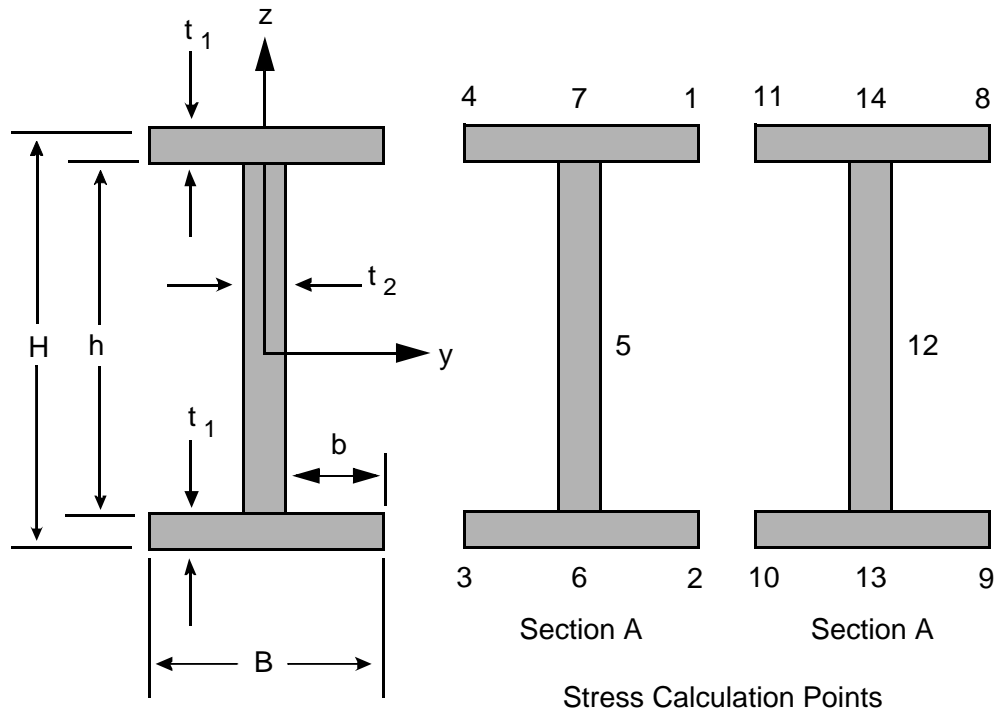


Figure 8-13

Design Variables:

Design Variable	Dimension
1	b
2	t_1
3	h
4	t_2

Other Dimensions:

$$H = h + 2t_1$$

$$B = 2b + t_2$$

Section Properties:

$$A = 2Bt_1 + (H - 2t_1)t_2 = A_1 + A_2$$

$$I_{yy} = \frac{BH^3 - (B - t_2)(H - 2t_1)^3}{12}$$

$$I_{zz} = \frac{2A_1B^2 + A_2t_2^2}{12}$$

$$J = \frac{2Bt_1^3 + ht_2^3}{3}$$

$$AS_y = \frac{20(1 + \nu)Bt_1}{12 + 11\nu}$$

$$AS_z = \frac{20(1 + \nu)Ht_2}{12 + 11\nu}$$

Stress Calculations:

Location	Component	Value
1, 8	σ_x	$-\frac{M_z B}{2I_{zz}} - \frac{M_y H}{2I_{yy}} + \frac{F_x}{A}$
2, 9	σ_x	$-\frac{M_z B}{2I_{zz}} + \frac{M_y H}{2I_{yy}} + \frac{F_x}{A}$
3, 10	σ_x	$\frac{M_z B}{2I_{zz}} + \frac{M_y H}{2I_{yy}} + \frac{F_x}{A}$
4, 11	σ_x	$\frac{M_z B}{2I_{zz}} - \frac{M_y H}{2I_{yy}} + \frac{F_x}{A}$
5, 12	τ_{xz}	$\frac{3F_z}{2t_2 H}$
6, 13	τ_{xy}	$\frac{3F_y}{4t_1 H}$
7, 14	τ_{xy}	$\frac{3F_y}{4t_1 B}$

Type 9: Rail

Description: This element is a frame element with the cross section described by the sizing variables b_1 , t_1 , b_2 , t_2 , h and t_3 . It is intended primarily for use in planar frame structures. Seven stresses are computed at each end of the element. These include four bending stresses and three shear stresses.

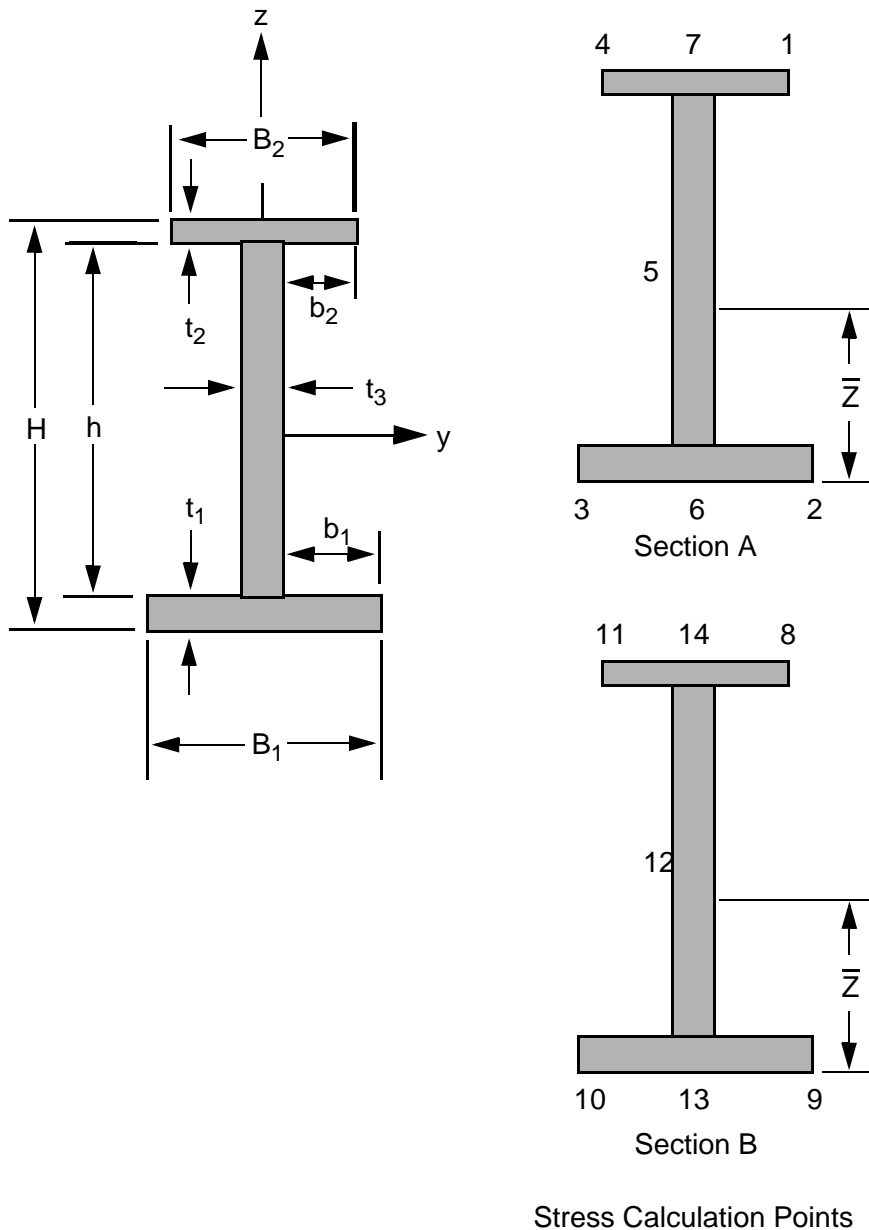


Figure 8-14

DVPROP3

Shape, Sizing, Topography, Topometry and Freeform Design Model Data

Design Variables:

Design Variable	Dimension
1	b_1
2	t_1
3	b_2
4	t_2
5	h
6	t_3

Other Dimensions:

$$H = h + t_1 + t_2$$

$$B_1 = 2b_1 + t_3$$

$$B_2 = 2b_2 + t_3$$

Section Properties:

$$A = B_1 t_1 + B_2 t_2 + (H - t_1 - t_2) t_3 = A_1 + A_2 + A_3$$

$$I_{yy} = \frac{A_1 t_1^2 + A_2 t_2^2 + A_3 (H - t_1 - t_2)^2}{12} + Q$$

$$I_{zz} = \frac{A_1 B_1^2 + A_2 B_2^2 + A_3 t_3^2}{12}$$

$$J = \frac{A_1 t_1^2 + A_2 t_2^2 + A_3 t_3^2}{3}$$

where

$$Q = A_1 \left(\bar{z} - \frac{t_1}{2} \right)^2 + A_2 \left(H - \bar{z} - \frac{t_2}{2} \right)^2 + A_3 (Z_1 - \bar{z})^2$$

$$\bar{z} = \frac{A_1 t_1 + A_2 (2H - t_2) + 2A_3 Z_1}{2A}$$

$$Z_1 = \frac{H - t_1 - t_2}{2} + t_1$$

$$AS_y = \frac{10(1 + \nu)(B_1 t_1 + B_2 t_2)}{12 + 11\nu}$$

$$AS_z = \frac{10(1 + \nu)H t_3}{12 + 11\nu}$$

Stress Calculations:

Location	Component	Value
1, 8	σ_x	$-\frac{M_z B_2}{2I_{zz}} - \frac{M_y(H - \bar{z})}{I_{yy}} + \frac{F_x}{A}$
2, 9	σ_x	$-\frac{M_z B_1}{2I_{zz}} + \frac{M_y \bar{z}}{I_{yy}} + \frac{F_x}{A}$
3, 10	σ_x	$\frac{M_z B_1}{2I_{zz}} + \frac{M_y \bar{z}}{I_{yy}} + \frac{F_x}{A}$
4, 11	σ_x	$\frac{M_z B_2}{2I_{zz}} - \frac{M_y(H - \bar{z})}{I_{yy}} + \frac{F_x}{A}$
5, 12	τ_{xz}	$\frac{3F_z}{2t_3 H}$
6, 13	τ_{xy}	$\frac{\alpha_1 3F_y}{2t_1 B_1}$
7, 14	τ_{xy}	$\frac{\alpha_2 3F_y}{2t_2 B_2}$

where;

$$\alpha_1 = \frac{I_1}{I_1 + I_2}, I_1 = \frac{t_1 B_1^3}{12} + \frac{(\bar{z} - t_1)t_3^3}{12}$$

$$\alpha_2 = \frac{I_2}{I_1 + I_2}, I_2 = \frac{t_2 B_2^3}{12} + \frac{(H - \bar{z} - t_2)t_3^3}{12}$$

Type 10: Tee

Description: The cross section is described by the sizing variables b , t_1 , h and t_2 . It is intended primarily for use in planar frame structures. Five stresses are computed at each end of the element. These include three bending stresses and two shear stress.

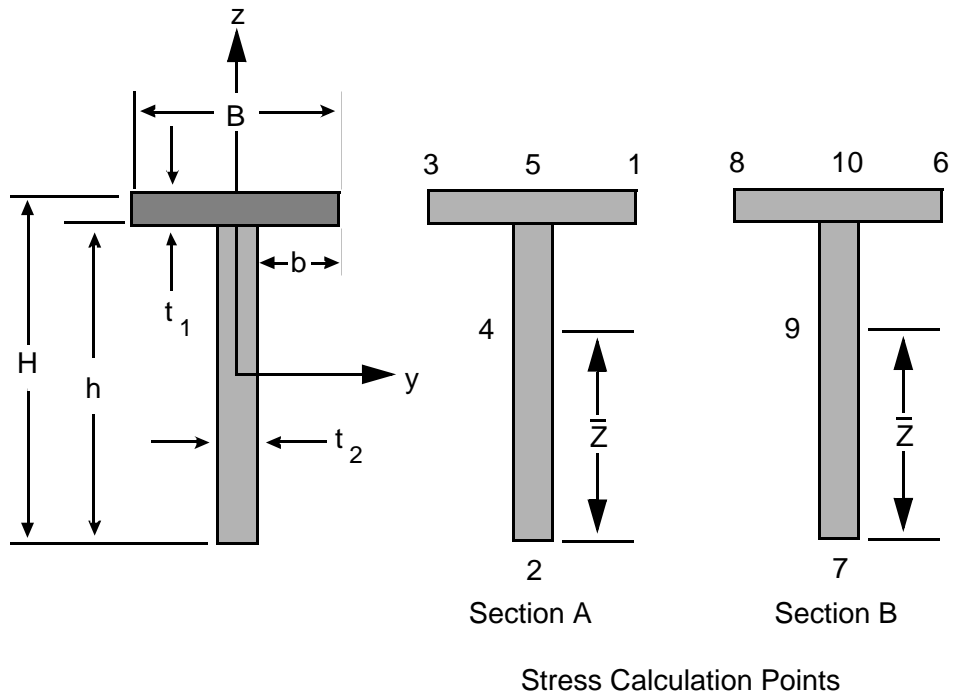


Figure 8-15

Design Variables:

Design Variable	Dimension
1	b
2	t_1
3	h
4	t_2

Other Dimensions:

$$H = h + t_1$$

$$B = 2b + t_2$$

Section Properties:

$$A = Bt_1 + ht_2 = A_1 + A_2$$

$$I_{yy} = \frac{Bt_1^3 + t_2(H - t_1)^3}{12} + Q$$

$$I_{zz} = \frac{A_1 B^2 + A_2 t_2^2}{12}$$

$$J = \frac{Bt_1^3 + t_2^3(H - t_1)}{3} = \frac{A_1 t_1^2 + A_2 t_2^2}{3}$$

where

$$Q = A_1 \left(H - \bar{z} - \frac{t_1}{2} \right)^2 + A_2 \left(\frac{H - t_1}{2} - \bar{z} \right)^2$$

$$\bar{z} = \frac{A_1(2H - t_1) + A_2(H - t_1)}{2A}$$

$$AS_y = \frac{10(1 + \nu)Bt_1}{12 + 11\nu}$$

$$AS_z = \frac{10(1 + \nu)Ht_2}{12 + 11\nu}$$

Stress Calculations:

Location	Component	Value
1, 6	σ_x	$-\frac{M_z B}{2I_{zz}} - \frac{M_y(H - \bar{z})}{I_{yy}} + \frac{F_x}{A}$
2, 7	σ_x	$\frac{M_y \bar{z}}{I_{yy}} + \frac{F_x}{A}$
3, 8	σ_x	$\frac{M_z B}{2I_{zz}} - \frac{M_y(H - \bar{z})}{I_{yy}} + \frac{F_x}{A}$
4, 9	τ_{xz}	$\frac{3F_z}{2t_2 H}$
5, 10	τ_{xy}	$\frac{3F_y}{2t_1 B}$

Type 11: Angle

Description: The cross section is described by the sizing variables b , t_1 , h and t_2 . Five stresses are computed at each end of the element. These include three bending stresses and two shear stresses. Warping of the section is not accounted for.

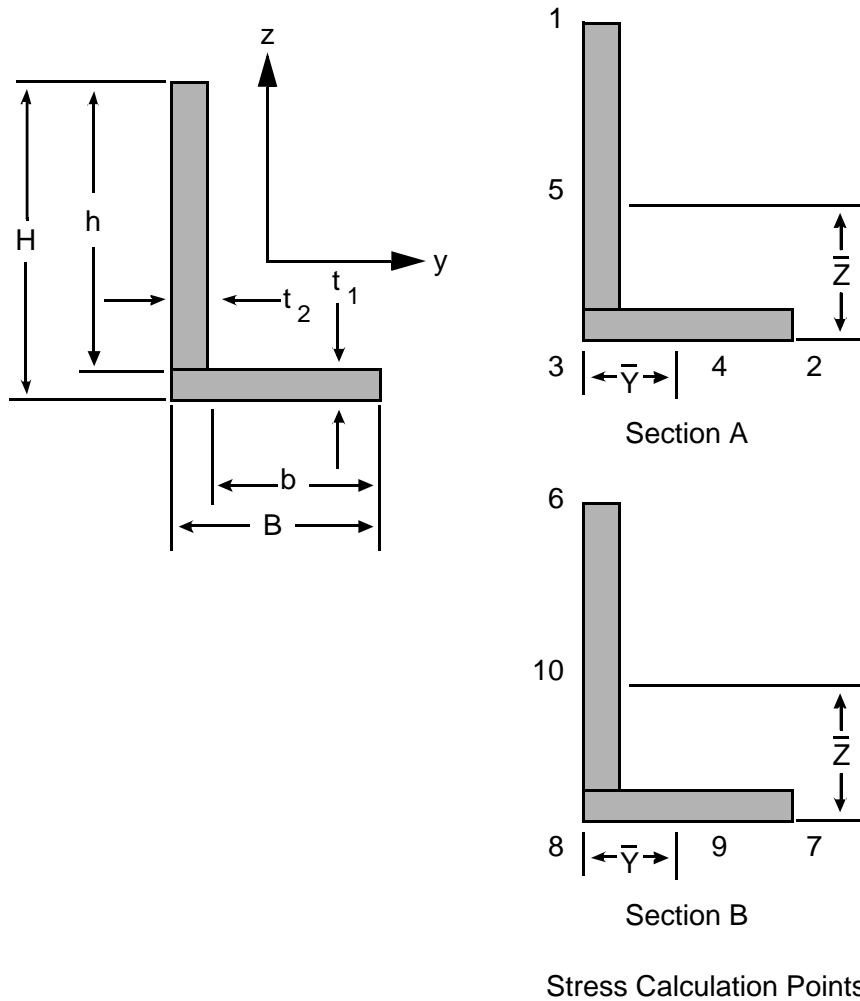


Figure 8-16

Design Variables:

Design Variable	Dimension
1	b
2	t_1
3	h
4	t_2

Other Dimensions:

$$H = h + t_1$$

$$B = b + t_2$$

Section Properties:

$$A = Bt_1 + ht_2 = A_1 + A_2$$

$$I_{yy} = \frac{Bt_1^3 + t_2(H-t_1)^3}{12} + Q$$

$$I_{zz} = \frac{t_1B^3 + (H-t_1)t_2^3}{12} + R$$

$$I_{yz} = \left(\bar{Y} - \frac{B}{2}\right)\left(\bar{z} - \frac{t_1}{2}\right)Bt_1 - \left(\bar{Y} - \frac{t_2}{2}\right)\left(\frac{H+t_1}{2} - \bar{z}\right)(H-t_1)t_2$$

$$J = \frac{Bt_1^3 + ht_2^3}{3}$$

where

$$Q = A_1\left(\bar{z} - \frac{t_1}{2}\right)^2 + A_2\left(\bar{z} - \frac{H+t_1}{2}\right)^2$$

$$\bar{z} = \frac{A_1t_1 + A_2(H+t_1)}{2A}$$

$$R = A_1\left(\bar{y} - \frac{B}{2}\right)^2 + A_2\left(\bar{y} - \frac{t_2}{2}\right)^2$$

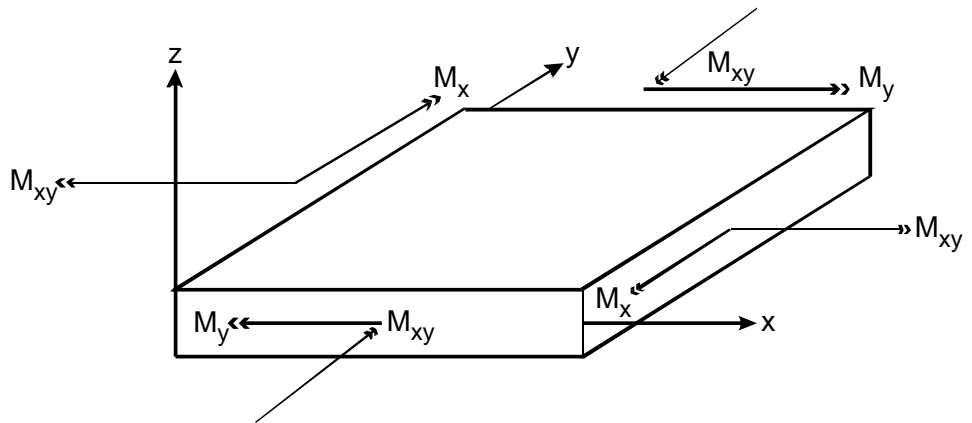
$$\bar{y} = \frac{A_1B + A_2t_2}{2A}$$

$$AS_y = \frac{10(1+\nu)Bt_1}{12+11\nu}$$

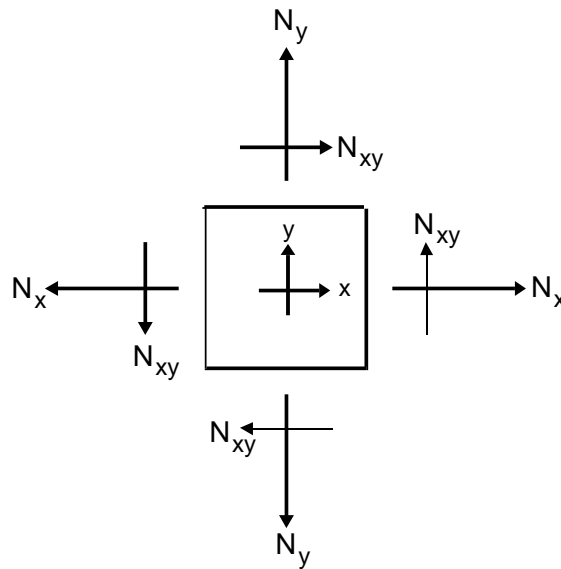
$$AS_z = \frac{10(1+\nu)Ht_2}{12+11\nu}$$

Stress Calculations:

Location	Component	Value
1, 6	σ_x	$\frac{(M_z I_{yy} - M_y I_{yz})\bar{y}}{I_{yy} I_{zz} - I_{yz}^2} - \frac{(M_y I_{zz} - M_z I_{yz})(H - \bar{z})}{I_{yy} I_{zz} - I_{yz}^2} + \frac{F_x}{A}$
2, 7	σ_x	$-\frac{(M_z I_{yy} - M_y I_{yz})(B - \bar{y})}{I_{yy} I_{zz} - I_{yz}^2} + \frac{(M_y I_{zz} - M_z I_{yz})\bar{z}}{I_{yy} I_{zz} - I_{yz}^2} + \frac{F_x}{A}$
3, 8	σ_x	$\frac{(M_z I_{yy} - M_y I_{yz})\bar{y}}{I_{yy} I_{zz} - I_{yz}^2} + \frac{(M_y I_{zz} - M_z I_{yz})\bar{z}}{I_{yy} I_{zz} - I_{yz}^2} + \frac{F_x}{A}$
4, 9	τ_{xy}	$\frac{3F_y}{2t_1 B}$
5, 10	τ_{xz}	$\frac{3F_z}{2t_2 H}$



(a) Plate element forces



(b) Membrane element

Figure 8-17

Type 12: Solid (Solid Plate)

Description: This is a solid plate element defined by design variable t. Stresses are calculated at the center of the top and bottom surfaces of the element.

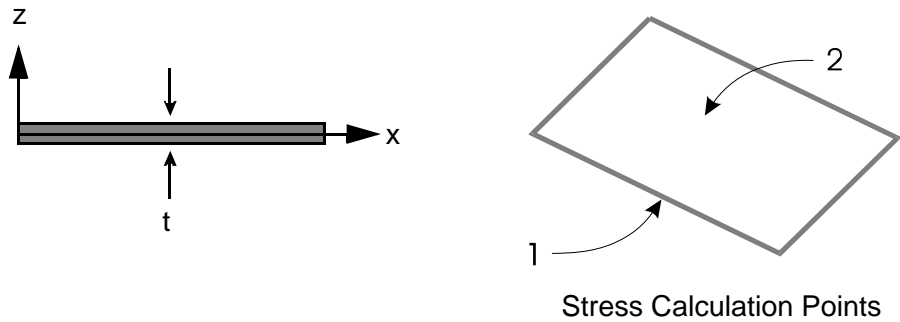


Figure 8-18

Design Variables:

Design Variable	Dimension
1	t

Section Properties:

$T = t$

$D = \frac{t^3}{12}$

$TS = \frac{5t}{6}$

Stress Calculations:

Item Number					Surf.	Comp.	Value
Static	Mag.	Phase	Real	Imag.			
1	-	-	-	-	1	τ_{\max}	Maximum Shear Stress
2	-	-	-	-	1	σ_{vm}	von Mises Stress
3	-	-	-	-	1	σ_1	Major Principal Stress
4	-	-	-	-	1	σ_2	Minor Principal Stress
5	5	19	33	47	1	σ_x	$\frac{N_x}{t} + \frac{M_x t}{2D}$
6	6	20	34	48	1	σ_y	$\frac{N_y}{t} + \frac{M_y t}{2D}$
7	7	21	35	49	1	τ_{xy}	$\frac{N_{xy}}{t} + \frac{M_{xy} t}{2D}$
8	-	-	-	-	2	τ_{\max}	Maximum Shear Stress
9	-	-	-	-	2	σ_{vm}	von Mises Stress
10	-	-	-	-	2	σ_1	Major Principal Stress
11	-	-	-	-	2	σ_2	Minor Principal Stress
12	12	26	40	54	2	σ_x	$\frac{N_x}{t} + \frac{M_x t}{2D}$
13	13	27	41	55	2	σ_y	$\frac{N_y}{t} + \frac{M_y t}{2D}$
14	14	28	42	56	2	τ_{xy}	$\frac{N_{xy}}{t} + \frac{M_{xy} t}{2D}$

For force and strain item numbers, see the DRESP1 data description.

Type 13: Sand (Sandwich Plate)

Description: This is a sandwich plate element defined by design variables t and h. Stresses are calculated at the center of the top and bottom surfaces of the element. The core has no bending stiffness.

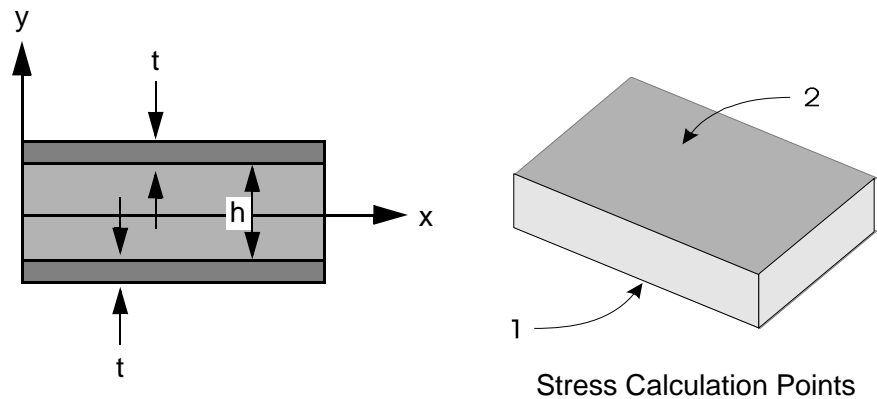


Figure 8-19

Design Variables:

Design Variable	Dimension
1	t
2	h

Section Properties:

$$T = 2t$$

$$D = \frac{(h + t)^2 t}{2} + \frac{t^3}{6}$$

$$TS = h$$

Stress Calculations:

Item Number					Surf.	Comp.	Value
Static	Mag.	Phase	Real	Imag.			
1	-	-	-	-	1	τ_{\max}	Maximum Shear Stress
2	-	-	-	-	1	σ_{vm}	von Mises Stress
3	-	-	-	-	1	σ_1	Major Principal Stress
4	-	-	-	-	1	σ_2	Minor Principal Stress
5	5	19	33	47	1	σ_x	$\frac{N_x}{T} + \frac{M_x z}{2D}$
6	6	20	34	48	1	σ_y	$\frac{N_y}{T} + \frac{M_y z}{2D}$
7	7	21	35	49	1	τ_{xy}	$\frac{N_{xy}}{T} + \frac{M_{xy} z}{2D}$
8	-	-	-	-	2	τ_{\max}	Maximum Shear Stress
9	-	-	-	-	2	σ_{vm}	von Mises Stress
10	-	-	-	-	2	σ_1	Major Principal Stress
11	-	-	-	-	2	σ_2	Minor Principal Stress
12	12	26	40	54	2	σ_x	$\frac{N_x}{T} + \frac{M_x z}{2D}$
13	13	27	41	55	2	σ_y	$\frac{N_y}{T} + \frac{M_y z}{2D}$
14	14	28	42	56	2	τ_{xy}	$\frac{N_{xy}}{T} + \frac{M_{xy} z}{2D}$

where $z = t + \frac{h}{2}$

For force and strain item numbers, see the DRESP1 data description.

Type 14: Sand2 (Two Thickness Sandwich Plate)

Description: This is a sandwich plate element defined by design variables t_1 , t_2 and h . Stresses are calculated at the center of the top and bottom surfaces of the element. The core has no bending stiffness. The bending plane is assumed to be in the same plane as corner grids. In other words, bending/membrane coupling does not exist.

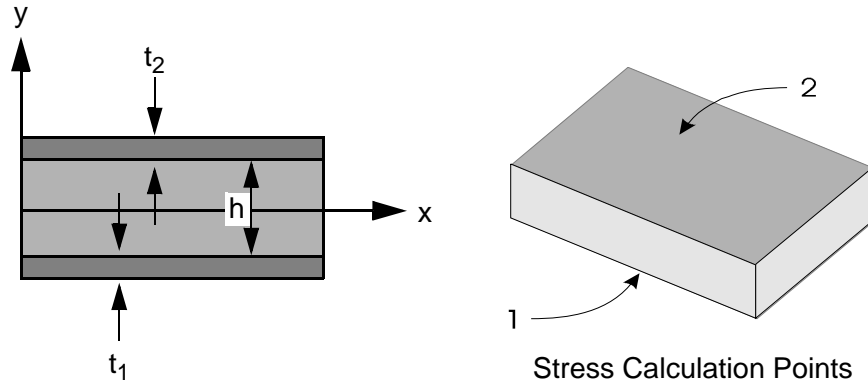


Figure 8-20

Design Variables:

Design Variable	Dimension
1	t_1
2	t_2
3	h

Section Properties:

$$T = t_1 + t_2$$

$$D = \left(\bar{x} - \frac{t_1}{2} \right)^2 t_1 + \left(t_1 + \frac{t_2}{2} + h - \bar{x} \right)^2 t_2 + \frac{t_1^3}{12} + \frac{t_2^3}{12}$$

$$\text{where } \bar{x} = \frac{\left[\frac{t_1 t_2}{2} + \left(t_1 + h + \frac{t_2}{2} \right) t_2 \right]}{t_1 + t_2}$$

$$TS = h$$

Stress Calculations:

Item Number					Surf.	Comp.	Value
Static	Mag.	Phase	Real	Imag.			
1	-	-	-	-	1	τ_{\max}	Maximum Shear Stress
2	-	-	-	-	1	σ_{vm}	von Mises Stress
3	-	-	-	-	1	σ_1	Major Principal Stress
4	-	-	-	-	1	σ_2	Minor Principal Stress
5	5	19	33	47	1	σ_x	$\frac{N_x}{T} + \frac{M_x z_1}{D}$
6	6	20	34	48	1	σ_y	$\frac{N_y}{T} + \frac{M_y z_1}{D}$
7	7	21	35	49	1	τ_{xy}	$\frac{N_{xy}}{T} + \frac{M_{xy} z_1}{D}$
8	-	-	-	-	2	τ_{\max}	Maximum Shear Stress
9	-	-	-	-	2	σ_{vm}	von Mises Stress
10	-	-	-	-	2	σ_1	Major Principal Stress
11	-	-	-	-	2	σ_2	Minor Principal Stress
12	12	26	40	54	2	σ_x	$\frac{N_x}{T} + \frac{M_x z_2}{D}$
13	13	27	41	55	2	σ_y	$\frac{N_y}{T} + \frac{M_y z_2}{D}$
14	14	28	42	56	2	τ_{xy}	$\frac{N_{xy}}{T} + \frac{M_{xy} z_2}{D}$

where $z_1 = -\bar{x}$ and $z_2 = t_1 + t_2 + h - \bar{x}$

For force and strain item numbers, see the DRESP1 data description.

Note: To include bending/membrane coupling effects, use PCOMP properties and DVPROP1/2 data statements.

8.2.30 DVPROP4

Data Entry: **DVPROP4** - Design Variable to Composite Properties Relations.

Description: Define multiple linear relations from design variables to composite ply thicknesses and angles.

Format:

1	2	3	4	5	6	7	8	9	10
DVPROP4	ID	PID		PMIN-T	DELP-T	DPMIN-T	PMIN-A	DELP-A	DPMIN-A
+	LABEL1	TC01	TDV1	TCOEF1	AC01.	ADV1	ACOEF1		
+	LABEL2	TC02	TDV2	TCOEF2	AC02	ADV2	ACOEF2		
+	-etc.-								

Example:

1	2	3	4	5	6	7	8	9	10
DVPROP4	130002	70049513							
+	layer1								
+	layer2	0.0	201	2.0		301			
+	layer3								
+	layer4	0.001	202	3.0		302			
+	layer5		203			303			

:

Field Information Description

2	ID	Unique identification number of the relation between the thicknesses/angles and design variables. (Integer > 0).
3	PID	PCOMP/PCOMPG property entry identification number. (Integer > 0).
5	PMIN-T	Minimum value allowed for the thickness. (Real or blank. Default = 1.0E-10)
6	DELP-T	Property (thickness) fractional move limit (Real > 0.0 or blank). (See Remark 3).
7	DPMIN-T	Property (thickness) minimum move limit factor (Real > 0.0 or blank). (See Remark 3).
5	PMIN-A	Minimum value allowed for the angles. (Real or blank. Default = -1.0E+35)
6	DELP-A	Property (angle) fractional move limit (Real > 0.0 or blank). (See Remark 3).
7	DPMIN-A	Property (angle) minimum move limit factor (Real > 0.0 or blank). (See Remark 3).

DVPROP4

Shape, Sizing, Topography, Topometry and Freeform Design Model Data

2	LABEL	Optional ply LABEL (Character or blank).
3	TC0i	Constant coefficient for thickness relation. (Real or blank. Default = 0.0).
4	TDVi	Thickness DVAR identification number. (Integer > 0 or blank).
5	TCOEFi	Linear coefficient for thickness relation. (Real or blank. Default = 1.0).
6	AC0i	Constant coefficient for angle relation. (Real or blank. Default = 0.0).
7	ADVi	Angle DVAR identification number. (Integer > 0 or blank).
8	ACOEFi	Linear coefficient for angle relation. (Real or blank. Default = 1.0).

Remarks

1. The thickness relationships are given by: $\text{Thickness}_i = \text{TC}_{0i} + \text{TDV}_i \times \text{TCOEF}_i$
2. The angle relationships are given by: $\text{Angle}_i = \text{AC}_{0i} + \text{ADV}_i \times \text{ACOEF}_i$
3. DELP and DPMIN default to 0.5 and 0.1 respectively, unless these values are specified in the **DOPT** entry (in which case they default to the values specified in DOPT). The minimum move limit value is defined as:
4. Minimum Move Limit Value =
$$\begin{cases} \text{DPMIN} & \text{if } |P_{\text{initial}}| \leq 1.0 \\ \text{DPMIN}|P_{\text{initial}}| & \text{if } |P_{\text{initial}}| > 1.0 \end{cases}$$
5. If any TDVi or ADVi is omitted, then the corresponding thickness or angle will not be designed by this entry.
6. The Z0, GE and non structural mass properties in PCOMPs cannot be designed using DVPROP4. If necessary, these properties can be designed using **DVPROP1** or **DVPROP2**.
7. DVPROP4 entries automatically generate equivalent DVPROP1 data. The sorted echo will show the generated DVPROP1 data.

8.2.31 DVSET

Data Entry: **DVSET** - Design Variable Discrete Value List

Description: Defines a set of discrete design variable values.

1	2	3	4	5	6	7	8	9	10
DVSET	SID	VAL1	VAL2	VAL3	VAL4	VAL5	VAL6	VAL7	VAL8
+	VAL9	VAL10	-etc.-						

Example:

1	2	3	4	5	6	7	8	9	10
DVSET	3	0.1	0.2	0.3	0.5	1.0	2.0	3.0	4.0
+	-0.1	-0.2							

Field	Information	Description
2	SID	Discrete design variable set identification number (Integer > 0).
3, 4, ...	VALi	Discrete design variable value (Real).

Remarks:

1. A discrete set can be referenced by any number of **DVAR** entries.
2. The discrete set values for a given SID will be the union of all DVSET and **DVSET1** entries using that set identification number.
3. Duplicate design variable values will be ignored.

8.2.32 DVSET1

Data Entry: **DVSET1** - Design Variable Discrete Value List, Alternate Form 1

Description: Defines a set of discrete design variable values.

Format:

1	2	3	4	5	6	7	8	9	10
DVSET1	SID	VAL1	VAL2	INCRE	NI				

Example 1 (Set = 1.0, 2.0, 3.0, 4.0, 5.0, 6.0, 7.0, 8.0, 9.0, 10.0):

1	2	3	4	5	6	7	8	9	10
DVSET1	3	1.0	10.0	1.0					

Example 2 (Set = 1.0, 2.0, 3.0, 4.0, 5.0, 6.0, 7.0, 8.0, 9.0, 10.0):

1	2	3	4	5	6	7	8	9	10
DVSET1	3	1.0		1.0	9				

Example 3 (Set = 1.0, 2.0, 3.0, 4.0, 5.0, 6.0, 7.0, 8.0, 9.0, 10.0):

1	2	3	4	5	6	7	8	9	10
DVSET1	3	1.0	10.0		9				

Field Information Description

2	SID	Discrete design variable set identification number (Integer > 0).
3	VAL1	First discrete design variable in set (Real).
4	VAL2	Last discrete design variable in set (Real > VAL2 or Blank).
5	INCRE	Discrete design variable increment (Real > 0.0 or Blank).
6	NI	Number of discrete design variable increments (Integer > 0 or Blank).

Remarks:

- Exactly two of VAL2, INCRE or NI must be non blank.
- When VAL1, VAL2 and INCRE are provided, the discrete values defined by this data entry are given by

$$VAL_i = VAL1 + (i-1)INCRE$$
, with $VAL_i \leq VAL2$. VAL2 will be included in the set.

3. When VAL1, INCRE and NI are provided, the discrete values defined by this data entry are given by
$$VAL_i = VAL1 + (i-1)INCRE \quad i = 1, (NI + 1).$$
4. When VAL1, VAL2 and NI are provided, the discrete values defined by this data entry are given by
$$VAL_i = VAL1 + (i-1)(VAL2-VAL1)/NI \quad i = 1, (NI + 1).$$
5. The discrete set values for a given SID will be the union of all **DVSET** and DVSET1 entries using that set identification number.
6. Duplicate design variable values will be ignored.

CHAPTER 9

Topology Design Model Data

- Topology Data Relationships
- Topology Bulk Data

9.1 Topology Data Relationships

The chart below shows the basic relationships among the data statements for design using *GENESIS* for topology optimization. It includes all topology design commands that may be included in the input data file. The chart also includes the most relevant analysis data that are referenced by the topology design data.

Bulk Data

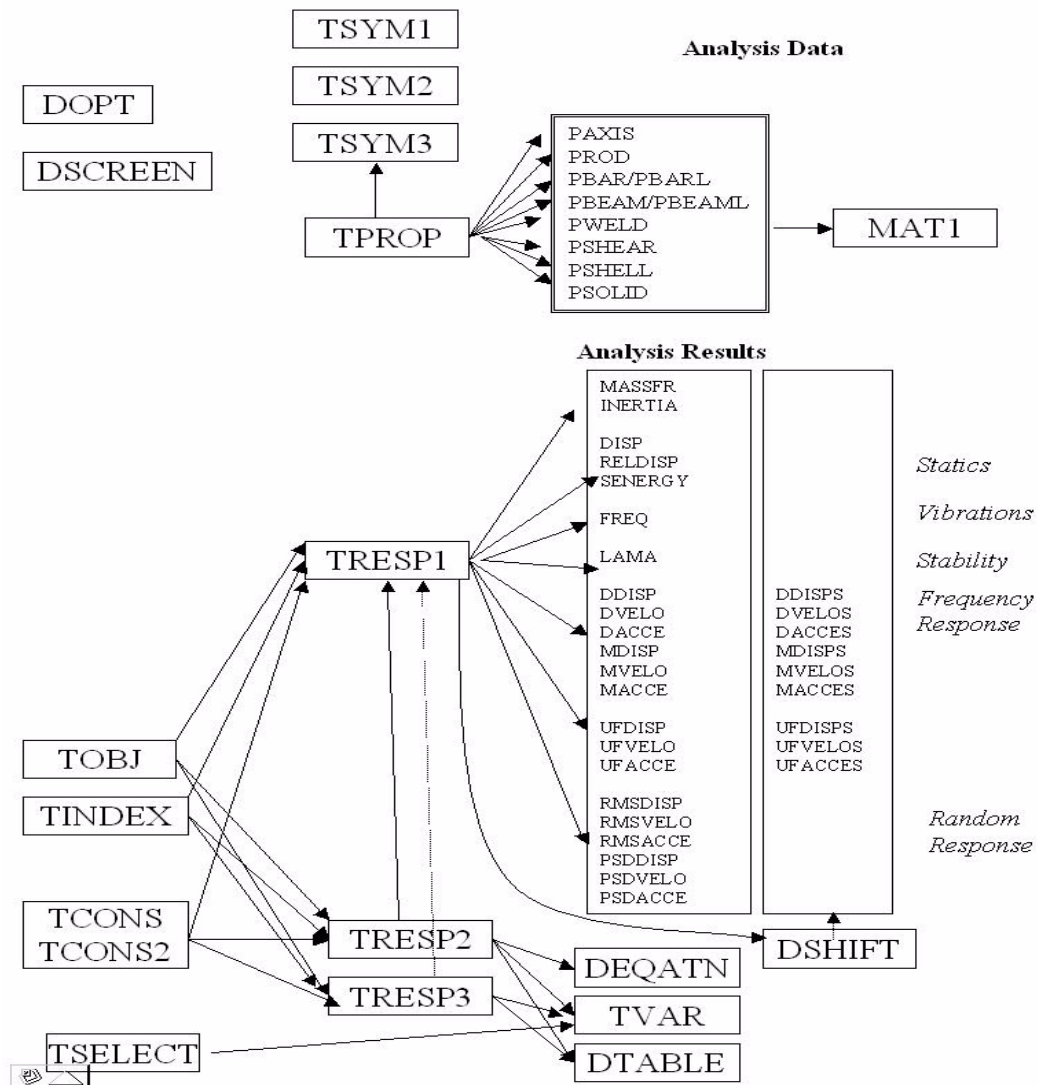


Figure 9-1

9.2 Topology Bulk Data

The bulk data for topology optimization using *GENESIS* is defined in this section. The data is given in alphabetical order.

9.2.1 DEQATN

Data Entry: **DEQATN** - Equation Application.

Description: Define equation.

Format:

1	2	3	4	5	6	7	8	9	10
DEQATN	EQID	EQUATION							
+	EQUATION (Cont.)								

Example 1:

1	2	3	4	5	6	7	8	9	10
DEQATN	2	F1(A,B,C,D,R) = A+B*C-(D**3+10.0)+SIN(3.14159*R)							
+	+A**2/(B-C)								

Example 2:

1	2	3	4	5	6	7	8	9	10
DEQATN	3	F(A1, A2, A3, A4, A5, A6, A7, A8, A9, A10, A11, A12, A13, A14, A15, A16, A17, A18,							
+	A19, A20) = A1+A2+A3+A4+A5+A6+A7+A8+A9+A10+A11+A12+A13+A14+A15+A16+A17+								
+	A18+A19+A20								

Example 3: Layered equation.

1	2	3	4	5	6	7	8	9	10
DEQATN	3	F(A1, A2, A3, A4, A5, A6, A7, A8, A9, A10) = A1+A2+A3; G=F+A4+A5+A6 ;							
+	H = F/G+A7+A8+A9+A10								

Field Information Description

2	EQID	Unique equation identification number. (Integer > 0).
3-...	EQUATION	Equation. See remarks.

Remarks:

1. DEQATN is also used for shape and sizing optimization. See **DEQATN** (p. 489).
2. See **Equation Utility** (p. 271) for a discussion of the user defined Equation feature.
3. EQUATION consists of the collection of data in fields 3 through 10 on the first entry and fields 2 through 10 on the continuations. The boundaries between these fields are not recognized and the collection is treated as if it were one field.
4. EQUATION may contain embedded blanks.

5. The left-hand side of the first equation must include the complete list of input parameters, enclosed in parentheses, used in the right-hand side of the equations. It must not include the names of the lower level equations.
6. The DEQATN entry is referenced by **TRESP2** data entries. The parameters in the left-hand side of the equation correspond sequentially to NDVi, NCj and (NRk-LIDRk) on TRESP2 entries.
7. The arithmetic operators in order of precedence are: ******, *****, **/**, **+**, **-**. The following relational operators may also be used: **=**, **/=**, **<**, **<=**, **>**, **>=**. The relational operators result in a value of 1.0 if the relation they are testing is true, or a value of 0.0 if the relation is false. The relational operators have lower precedence than the arithmetic operators.
8. The following table lists the available intrinsic functions:

Function	Description
ABS(x)	Absolute value of x
ACOS(x)	Inverse cosine of x (result in radians)
ACOSH(x)	Inverse hyperbolic cosine of x
ASIN(x)	Inverse sine of x (result in radians)
ASINH(x)	Inverse hyperbolic sine of x
ATAN(x)	Inverse tangent of x (result in radians)
ATAN2(y,x)	Inverse tangent of y/x (result in radians, -π to π)
ATANH(x)	Inverse hyperbolic tangent of x
ATANH2(y,x)	Inverse hyperbolic tangent of y/x
AVG(x1,x2,...,xn)	Average: $(x1+x2+...+xn)/n$
COS(x)	Cosine of x (x in radians)
COSH(x)	Hyperbolic cosine of x
COTAN(x)	Cotangent of x (x in radians)
DIM(x,y)	Maximum of (0, x-y)
EXP(x)	e raised to power x
INT(x)	Convert x to integer
LOG(x)	Natural (base e) logarithm of x
LOG10(x)	Common (base 10) logarithm of x
LOGX(x,y)	Base x logarithm of y
MAX(x1,x2,...,xn)	Maximum of (x1, x2, ..., xn)
MIN(x1,x2,...,xn)	Minimum of (x1, x2, ..., xn)
MOD(x,y)	Remainder of x/y

Function	Description
PI(x)	π times x
RSS(x1,x2,...,xn)	Square root of sum of squares: $\text{SQRT}(x1^{**2} + x2^{**2} + \dots + xn^{**2})$
SIGN(x,y)	Absolute value of x times sign of y
SIN(x)	Sine of x (x in radians)
SINH(x)	Hyperbolic sine of x
SQRT(x)	Square root of x
SSQ(x1,x2,...,xn)	Sum of squares: $(x1^{**2} + x2^{**2} + \dots + xn^{**2})$
SUM(x1,x2,...,xn)	Sum: $(x1 + x2 + \dots + xn)$
TAN(x)	Tangent of x (x in radians)
TANH(x)	Hyperbolic tangent of x

9. The relational operators and the functions ABS, DIM, INT, MAX, MIN, MOD, and SIGN should be used with caution, because they can create discontinuities in the function or its derivative. Such discontinuities can cause poor convergence behavior of the optimizer.
10. The maximum number of characters that can be used to define the function names and each of the arguments is 31.
11. Layered equations can be used by separating the equations with semi colons (;). The first equation contains the argument list for all the equations. Equations may reference the value of one or more preceding equations. The value of the last equation is the result of the DEQATN and is the value used by the TRESP2 entry.
12. Constants specified in **DTABLE** data can be used in DEQATN data without passing them through the argument list. If the constant name on DTABLE is the same as an argument list or function name, then the constant in DTABLE is not used in the equation.

9.2.2 DOPT

Data Entry: **DOPT** - Optimization Parameters.

Description: Define parameters to be used in optimization.

Format:

1	2	3	4	5	6	7	8	9	10
DOPT	DESMAX								
+	NAM1	VAL1	NAM2	VAL2	NAM3	VAL3	NAM4	VAL4	
+	NAM5	VAL5	-etc.-						

Example:

1	2	3	4	5	6	7	8	9	10
DOPT	10								
+	CONV1	0.0001	IPRINT	3	CONVDV	0.01			

Field Information Description

2	DESMAX	Maximum number of approximate optimizations to be performed. (Integer > 0 or blank, Default = 15).
2,4,...	NAMi	Name of additional parameter (see table below for available parameters).
3,5,...	VALi	Value of additional parameter. Real or Integer as indicated by the table below.

Remarks:

1. Only one DOPT statement may appear in the input data file.
2. DOPT is also used for shape and sizing optimization. See **DOPT** (p. 507).

Available topology optimization parameters are listed below.

Method Switches

PARAMETER	POSSIBLE VALUES	DEFAULT	DEFINITION
LINAPR	0 1 -1	0	Approximation Optimization Method Control LINAPR = 0 will cause the program to use the “fast” linear approximations method when all approximations are linear. If any approximations are nonlinear, then the program will use its standard hybrid approximations. LINAPR = 1 will force the program to use the “fast” linear approximations method for all responses. This will result in faster times in each design cycle, but it might lower the quality of approximations which in some cases might cause the program to need more design cycles to converge. LINAPR = -1 will cause to use its standard linear and hybrid approximations.
METHOD	1 2 3	1	Optimization method to be used by DOT. METHOD = 1 means use the modified method of feasible directions. Method = 2 means use sequential linear programming. Method = 3 means use sequential quadratic programming. If the problem is unconstrained (NCON=0), the BFGS algorithm will be used if METHOD=1 and the Fletcher-Reeves algorithm will be used if METHOD=2.
OPTM	0 1	1	If OPTM = 0, <i>GENESIS</i> will use the DOT optimizer. If OPTM = 1, <i>GENESIS</i> will use the BIGDOT optimizer. Default = 0 for shape and sizing and 1 for topology.
TINDEXM	0 1 2 3	0	Compliance index formulation. A value of 0 will cause <i>GENESIS</i> to use the reciprocal contribution for responses with negative weighting factors. Responses are normalized. A value of 1 will cause <i>GENESIS</i> to use the direct contribution for responses with negative weighting factors. Responses are normalized. A value of 2 will cause <i>GENESIS</i> to use the reciprocal contribution for responses with negative weighting factors. A value of 3 will cause <i>GENESIS</i> to use the direct contribution for responses with negative weighting factors. If shape/sizing data is mixed with topology data, TINDEXM must have the same value as DINDEXM .

TCYCLEM	0 1 -1, -2, -3, -4, -5, -6	0	A value of 0 will cause <i>GENESIS</i> to ignore the progressive rule. A value of 1 will cause <i>GENESIS</i> to use the progressive rule. A negative value will cause <i>GENESIS</i> to use a built-in progressive rule.
DESMAXM	0 1	*	A value of 1 will cause <i>GENESIS</i> to override DESMAX with the number of design cycles needed to complete the progressive rule schedule. A value of 0 will cause <i>GENESIS</i> to use DESMAX as the maximum number of design cycles, which means the entire progressive rule schedule may not complete. If the DESMAX field is not blank, the default for DESMAXM is 0. If DESMAX is blank, then the default is 1. If TCYCLEM is not 1, then DESMAXM is ignored.
FILTER	-2 -1 0 1	1	Anti-checkerboard filtering ON (FILTER \neq 0) or OFF (FILTER = 0) parameter. If FILTER=1, <i>GENESIS</i> will use a global anti-checkerboard filter region. If FILTER=-1, <i>GENESIS</i> will use property-by-property anti-checkerboard filter regions. If FILTER=-2, <i>GENESIS</i> will use property-by-property anti-checkerboard filter regions and it will take in consideration the angle of the norms of shell elements to determine the neighbor influences. The more closely aligned neighbor element norms are, the stronger those elements will interact in the filter. Orthogonal elements will not interact. For non-shell elements, FILTER = -2 is treated the same as FILTER = -1. Parameter FILTNRM can be used with FILTER=-2 to further limit the number of elements that interact
FILTNRM	$0.0 \leq$ Real ≤ 90.0	90.0	Cutoff angle to exclude neighbor elements from participating in an anti-checkerboard filter region. In nonplanar structures, smaller values of FILTNRM reduces the number of elements in a checkerboarder reggion. Conversely, larger values of FILTNRM increases the number of elements in checkerboarder region. A value of 0.0 will force <i>GENESIS</i> to create anti-checkerboard filter regions containing only neighbor elements in the same plane. A value of 90.0 will let <i>GENESIS</i> include all neighbor elements (however elements close to 90 degree will have little to no influence as a result of FILTER=-2). This parameter is only used used with FILTER =-2 and it is ignored with other types. This parameter is only used used with shell elements and it is ignored with other types.

POLEM	0 1	0	<p>A value of 1 will cause <i>GENESIS</i> to choose the hybrid based topology method over the geometry based topology method. A value of 0 will cause <i>GENESIS</i> to choose the geometry based topology method over the hybrid based topology method.</p> <p>The geometry or hybrid topology methods are available when minimum member size is used in TSYM1, TSYM2 or TSYM3. If minimum member size is not used, then the element based topology method is used and this parameter is ignored.</p>
TOPDIS	$0.0 \leq$ Real ≤ 1.0	0.0	<p>A value greater than 0.0 will cause <i>GENESIS</i> to perform an extra final design cycle where all topology design variables are fully polarized. Topology design variables with values greater than or equal to <i>TOPCUT</i> will be set to 1.0, while all other topology variables will be set to their lower bound (usually 0.0).</p> <p>If the DOPT parameter TOPDISM is 0 (or blank), then <i>TOPCUT</i> is TOPDIS. If TOPDISM is 1 then <i>TOPCUT</i> = TOPDIS**(1/RV1), where RV1 is the (final) power.</p>
TOPDISM	0 1	0	<p>TOPDISM parameter to decide wheater TOPDIS represent densities or Young's modulus fraction.</p> <p>A value of 0 of TOPDISM will cause <i>GENESIS</i> to use TOPDIS as the density cutoff value.</p> <p>A value of 1 of TOPDISM will cause <i>GENESIS</i> to use TOPDIS as a Young's modulus cuttoff value.</p>

Parameters for Hard Convergence

PARAMETER	POSSIBLE VALUES	DEFAULT	DEFINITION
CONV1	Real > 0.0	0.001	Relative change criterion to detect hard convergence of the overall optimization process. Terminate if the <u>relative</u> change in the <u>objective function</u> is less than CONV1 for two consecutive design cycles and all constraints are satisfied within a tolerance of GMAX.
CONV2	Real > 0.0	0.001	Absolute change criterion to detect hard convergence of the overall optimization process. Terminate if the <u>absolute</u> change in the <u>objective function</u> is less than CONV2 for two consecutive design cycles and all constraints are satisfied within a tolerance of GMAX. $\text{CONV2} = \text{Max}(\text{CONV2} * \text{OBJ}_{\text{initial}} , 1.0\text{E-}19)$
GMAX	Real > 0.0	0.005	Maximum <u>constraint</u> violation allowed at the optimum. Constraints are normalized so a value of 0.01 represents a one percent constraint violation, which is normally considered acceptable.
DELOBJ	Real > 0.0	0.00001	Relative change criterion to detect hard convergence of the approximate optimization realized by BIGDOT. Terminate if the <u>relative</u> change in the <u>objective function</u> is less than DELOBJ for ITRMOP consecutive iterations and all constraints are satisfied within a tolerance.
DABOBJ	Real > 0.0	0.00001	Absolute change criterion to detect hard convergence of the approximate optimization process realized by BIGDOT. Terminate if the <u>absolute</u> change in the <u>objective function</u> is less than DABOBJ for ITRMOP consecutive iterations and all constraints are satisfied within a tolerance.

Parameters for Soft Convergence

PARAMETER	POSSIBLE VALUES	DEFAULT	DEFINITION
CONVCN	$0.0 \leq \text{Real}$	0.001	Allowable change in the maximum <u>constraint</u> value for soft convergence. If the change in the maximum constraint value is less than CONVCN, and CONVDV is satisfied, then terminate the design process with soft convergence.
CONVDV	Real > 0.0	0.0001	Relative change criterion to detect soft convergence of the overall optimization process. Terminate with soft convergence if the maximum relative change in the <u>design variables</u> is less than CONVDV during the approximate optimization and CONVCN is satisfied.

Parameters for Grey Element Convergence

PARAMETER	POSSIBLE VALUES	DEFAULT	DEFINITION
CONVTS	$0.0 \leq \text{Real}$	0.0	<p>Allowable change in the <u>grey fraction</u> for grey convergence. If all constraints are satisfied and the change in the final <u>grey fraction</u> value between two consecutive stages is less than CONVTS, then terminate the design process with grey convergence.</p> <p>This parameter is only used with the progressive rule. Grey fraction is the number of elements with intermediate density values divided by the total number of topology designable elements.</p>
CONVTD	$0.0 \leq \text{Real}$	0.0	<p>Allowable change in the <u>grey fraction</u> for grey convergence. If the change in the <u>grey fraction</u> value in two consecutive design cycles, occurring in the last stage of the progressive schedule, is less than CONVTD, and CONVCN is satisfied, then terminate the design process with grey convergence.</p> <p>This parameter is only used with the progressive rule. Grey fraction is the number of elements with intermediate density values divided by the total number of topology designable elements.</p>

Parameters for Contact Analysis Optimization

PARAMETER	POSSIBLE VALUES	DEFAULT	DEFINITION
CNTDC	Integer > 0	1	<p>Defines in which design cycles the program will perform full contact analysis. A value of n means that full contact analysis is performed every n-th design cycle. Regardless of the value of CNTDC, the program will always perform full contact analysis in the first and last design cycles.</p> <p>A value of 1, the default, causes the program to perform full contact analysis in every design cycle.</p> <p>A value greater than 1 will allow the program to use unconverged contact analyses in the skipped design cycles. The maximum number of contact iterations that the program will perform for the skipped design cycles is defined by the parameter CNTIT. For more information, See Nonlinear Contact Analysis Optimization (p. 28)</p>
CNTIT	Integer > 0	1	<p>Defines a maximum number of contact iterations that the program will perform in the design cycles that are not performing full converged contact analyses. This parameter is only used when CNTDC > 1.</p>

Information Switches

PARAMETER	POSSIBLE VALUES	DEFAULT	DEFINITION
IPRINT	Integer [0-7]	0	Print control for optimizer output during approximate optimization. Larger value gives more print. IPRINT=0 gives no print.
TEMLL	$0.0 \leq \text{Real} < 1.0$	0.25	Controls the count of topology designed elements with low Young's modulus values. Elements whose Young's modulus values are lower or equal to TEMLL*Initial Young's modulus will be counted as low modulus elements.
TEMUL	$0.0 \leq \text{Real} < 1.0$	0.75	Control the count of topology designed elements with high Young's modulus values. Elements whose Young's modulus values are higher or equal to TEMUL*Initial Young's modulus will be counted a high modulus elements.

Topology Move Limit Parameters

PARAMETER	POSSIBLE VALUES	DEFAULT	DEFINITION
DELT	Real > 0.0	1.0E-6	Fractional change allowed for each designable topology variable during the approximate optimization. This provides move limits.
DTMIN	Real > 0.0	0.2	Minimum move limit fraction imposed for topology design variables.
TMIN	$0.0 \leq \text{Real} < 1.0$	0.0	Minimum value allowed for the topology design variable.
DELX	Real > 0.0	0.5	Fractional change allowed for each topology extra variable (TVAR) during the approximate optimization.
DXMIN	Real > 0.0	0.1	Minimum move limit factor imposed for topology extra variables (TVAR). The minimum move limit is defined as: $\text{MinimumMoveLimit} = \begin{cases} \text{DXMIN} & \text{if } X_{\text{initial}} \leq 1.0 \\ \text{DXMIN} X_{\text{initial}} & \text{if } X_{\text{initial}} > 1.0 \end{cases}$

File Printing Controls

PARAMETER	POSSIBLE VALUES	DEFAULT	DEFINITION
DNSHIS	0 1	0	A value of 1 will cause the program to print the density *.DNS" file. A value of 0 will cause the program to not print the density *.DNS" file.

Other Parameters

PARAMETER	POSSIBLE VALUES	DEFAULT	DEFINITION
DESMIN	Integer > -1	0	Minimum number of design cycles. A value n, greater than zero, will force the program to perform at least n design cycles before stopping.
ITMAX	Integer > 0	100	Maximum number of iterations in the approximate optimization.
ITRMOP	Integer > 0	3	The number of consecutive iterations DOT or BIGDOT must satisfy the absolute or relative convergence criteria before optimization is terminated. Usually ITRMOP should be at least 2 because it is common to make little progress on one iteration, only to make major progress on the next. Therefore, ITRMOP = 2 will allow a second try before terminating. If progress toward the optimum seems slow, but consistent, and function evaluations are not too expensive, it may improve the solution to increase ITRMOP to a value of 3 to 10.
DVINIT	-1 0 1	0	A value of -1 will cause <i>GENESIS</i> to reset the initial value of all topology extra variables to their lower bound. A value of 1 will cause <i>GENESIS</i> to set all topology extra variables to their upper bounds. A value of 0 will leave the topology extra variables at the original initial value.
BDMEM	$0.0 \leq \text{Real} \leq 100.0$	0.0	Percentage of free memory in the approximate module that will be additionally reserved for BIGDOT.

FREQREG	Integer $\neq 0$	1	<p>Define how responses from different loading frequencies are grouped into screening regions.</p> <p>If FREQREG > 0 then FREQREG corresponds to the maximum number of dynamic loading frequencies per region.</p> <p>If FREQREG < 0 then ABS(FREQREG) corresponds to the maximum number of regions per frequency response loadcase.</p> <p>The most conservative screening method corresponds to FREQREG=1 and the most aggressive screening method corresponds to FREQREG=-1.</p> <p>An aggressive choice can cause the program to retain fewer responses, which will reduce sensitivity calculation time. However an aggressive choice may cause the program to need more design cycles to converge.</p>
RESETMV	Integer $\neq 0$	*	<p>Defines when or how often move limits are restored to their original values.</p> <p>If RESETMV > 0 then RESETMV corresponds to the design cycle number on which move limits are reset. This is a one time event. If the program converged before RESETMV number of design cycles, then this parameter is not used effect.</p> <p>If RESETMV < 0 then ABS(RESETMV) corresponds to the frequency on which move limits are reset. If N is the frequency of resetting, then the program will reset the move limits every N design cycles.</p> <p>The default will cause the program to not reset move limits in a given run.</p> <p>Note: move limits can be changed by automatic move limits adjustments. It should be noted that restarting the program also reset the move limits starting from the restarted design cycle.</p>
TPQVOL	0 1	1	<p>Defines whether or not internal heat generation fluxes for heat transfer analysis are scaled by topology design variables.</p> <p>If TPQVOL = 1, the default, QVOL fluxes are scaled by topology variables.</p> <p>If TPQVOL = 0, QVOL fluxes are not dependent on topology variables. This means that even if an element's conductivity goes away due to topology, that element can still have internal heat generation fluxes.</p>

9.2.3 DSCREEN

Data Entry: **DSCREEN** - Constraint Screening Data.

Description: Define constraint screening data for constraint deletion.

Format:

1	2	3	4	5	6	7	8	9	10
DSCREEN	TYPE	TRS	NSTR						

Example:

1	2	3	4	5	6	7	8	9	10
DSCREEN	DISP	-0.5							

Field	Information	Description
-------	-------------	-------------

2	TYPE	Type of the constraints for which the screening criteria apply. Must be DISP, FORCE, STRESS, FREQ, EQUA or SENERGY
3	TRS	Truncation threshold. (Real \leq 0.0 or blank, Default = -0.5).
4	NSTR	Maximum number of constraints to be retained per region per load case (per loading frequency). (Integer > 0 or blank) (Default = 20).

Remarks:

- DSCREEN is also used for shape and sizing optimization, and different options are available there. See **DSCREEN** (p. 570).
- Constraints are retained if

$$\frac{\text{Response} - \text{UBi}}{|\text{UBi}|} \geq \text{TRS}$$
 or

$$\frac{\text{LBi} - \text{Response}}{|\text{LBi}|} \geq \text{TRS}$$
 where Response is defined on the TRESPi entry and LBi and UBi are defined on the TCONS entry. If |UBi| is zero, the above equation containing |UBi| is replaced by Response \geq TRS. Similarly, when |LBi| is zero, the equation containing |LBi| is replaced by Response \leq -TRS.
- For DDISP, DVELO, DACCE, DDISPS, DVELOs, DACCES, MDISP, MVELO, MACCE, MDISPS, MVELOs, MACCES, PSDDISP, PSDVELO, PSDACCE, PSDDS, PSDVS, PSDAS, RMSDISP, RMSVELO, RMSACCE, and EVECT, use DISP constraint screening data.
- For CPRESS use STRESS constraint screening data.
- For SPCF use FORCE constraint screening data.

6. The DOPT parameter **FREQREG** can be used to control how responses from different loading frequencies in a frequency response loadcase are grouped into regions.

9.2.4 DSHIFT

Data Entry: - Frequency Response Shifting Definition.

Description: Defines a transformation to shift or scale dynamic displacement, velocity or acceleration responses by a function of the loading frequency.

Format:

1	2	3	4	5	6	7	8	9	10
DSHIFT	ID	COEFF	FTYPE	REFVAL	TID	ETYPE			

Example 1: Create a shifted response, where the tabled values are used to normalize the response. Assume that a TRESP1 that reference MACCES points to a 1001

$$S(f) = \frac{20 \log_{10} \left(\frac{ACCE(f)}{0.02} \right)}{T(f)} \quad \therefore$$

1	2	3	4	5	6	7	8	9	10
DSHIFT	1001	20.0	LOG10	0.02	20001	2			

Example 2: Create a shifted response, where the tabled values are subtracted from the response. Assume that a TRESP1 that references MACCES points to 1002

$$.S(f) = 20 \log_{10} \left(\frac{ACCE(f)}{0.02} \right) - T(f) :$$

1	2	3	4	5	6	7	8	9	10
DSHIFT	1002	20.0	LOG10	0.02	20001	1			

Format:

Field Information Description

2	ID	DSHIFT identification number (Integer > 0).
3	COEFF	Scale factor (Real or blank. Default = 1.0).
4	FTYPE	Function type. One of "IDENT", "LOG10", "LOG", "RECIP" or Blank. Default is "IDENT". See remark 3.
5	REFVAL	Reference value. This value divides the frequency response H(f) associated to the TRESP1 that references . (Real > 0.0 or blank. Default = 1.0).
6	TID	Identification number of a TABLED1, TABLED2, TABLED3 or TABLED4 entry which defines T(f) (Integer > 0).
7	ETYPE	Equation type. (Integer 1 or 2). If ETYPE = 1: TABLEDi values are subtracted. If ETYPE = 2: TABLEDi values are used for normalization.

Remarks:

- 1. entries are used to specify parameters to create customized responses from frequency response loadcases that are transformed by a user-specified function of the loading frequency. Shifted responses are function of the dynamic displacement, velocities and accelerations that are listed in **TRESP1**, and are created by listing DDISPS, DVELOs, DACCES, MDISPS, MVELOs, MACCES, PSDDS, PSDVS, PSDAS, UFDISPS, UFVELOs, UFACCES or ERPS in TRESP1 and referencing the data.
- 2. Shifted responses can be used to create a design constraint in which the bound is a function of the loading frequency. See **Loading Frequency Dependent Constraint Bounds** (p. 320)
- 3. In the following tables, $H(f)$ represents the dynamic displacement, velocity or acceleration magnitude of the grid and component specified on the referencing TRESP1. For FTYPE = IDENT, the following transformations are used:

	FTYPE = IDENT
ETYPE = 1	$S(f) = \text{Coeff} \times \frac{H(f)}{\text{Refval}} - T(f)$
ETYPE = 2	$S(f) = \frac{\text{Coeff} \times \frac{H(f)}{\text{Refval}}}{T(f)}$

For FTYPE = LOG10, the following transformations are used

	FTYPE = LOG10
ETYPE = 1	$S(f) = \text{Coeff} \times \log_{10}\left(\frac{H(f)}{\text{Refval}}\right) - T(f)$
ETYPE = 2	$S(f) = \frac{\text{Coeff} \times \log_{10}\left(\frac{H(f)}{\text{Refval}}\right)}{T(f)}$

For FTYPE = LOG, the following transformations are used

	FTYPE = LOG
ETYPE = 1	$S(f) = \text{Coeff} \times \ln\left(\frac{H(f)}{\text{Refval}}\right) - T(f)$
ETYPE = 2	$S(f) = \frac{\text{Coeff} \times \ln\left(\frac{H(f)}{\text{Refval}}\right)}{T(f)}$

For FTYPE = RECIP, the following transformations are used:

	FTYPE = RECIP
ETYPE = 1	$S(f) = \text{Coeff} \times \frac{\text{Refval}}{H(f)} - T(f)$
ETYPE = 2	$S(f) = \frac{\text{Coeff} \times \frac{\text{Refval}}{H(f)}}{T(f)}$

4. This entry is also used for shape and sizing optimization. See [DSHIFT](#) (p. 579).

9.2.5 DTABLE

Data Entry: **DTABLE** - Table Constants.

Description: Define a table of constants that are frequently used in equations.

Format:

1	2	3	4	5	6	7	8	9	10
DTABLE	LABL1	VALU1	LABL2	VALU2	LABL3	VALU3	LABL4	VALU4	
+	LABL5	VALU5	LABL6	VALU6	-etc.-				

Example:

1	2	3	4	5	6	7	8	9	10
DTABLE	PI	3.1416	E	1.0E+6	RHO	0.1	NU	0.3	
+	BK	40.	ALPHA	0.001					

Field Information Description

2,4,..	LABLi	Unique identification label for the constant (non blank characters).
3,5,..	VALUi	Value of the constant (Real).

Remarks:

1. DTABLE is also used for shape and sizing optimization. See **DTABLE** (p. 588).
2. DTABLE data is referenced by **TRESP2** to be used with **DEQATN** or by **TRESP3** to be used in the user-subroutine.
3. Multiple DTABLE entries may appear in the input data.
4. VALUi is referenced by using the corresponding LABLi for NCi on the TRESP2 or TRESP3 data.
5. As an alternative to passing the tabled values through the argument list from TRESP2, the table constant name LABLi can be used directly in DEQATN data in the equation to represent the constant VALUi. In this case, the program will substitute VALUi for LABLi as long as LABLi is distinct from the argument names and function names.

9.2.6 TCONS

Data Entry: **TCONS** - Topology Constraints.

Description: Define topology constraints.

Format:

1	2	3	4	5	6	7	8	9	10
TCONS	RID	LID1	LB1	UB1	LID2	LB2	UB2		
+		LID3	LB3	UB3	-etc.-				

Example:

1	2	3	4	5	6	7	8	9	10
TCONS	16	10	0.5	25.0					

Field Information Description

2	RID	TRESP1 , TRESP2 or TRESP3 identification number (Integer > 0).
3,6	LIDi	Load case identification number (Integer ≥ 0 or ALL or blank. Default = ALL).
4,7	LBi	Constraint lower bound imposed on this response quantity. (Real or blank. Default = -1.0E30).
5,8	UBi	Constraint upper bound imposed on this response quantity. (Real or blank, $LB \leq UB$. Default = 1.0E30).

Remarks:

1. A TRESP1, TRESP2 or TRESP3 entry can be referred to by at most one TCONS entry. A TCONS entry cannot reference a TRESP1, TRESP2 or TRESP3 that is referred to by a **TCONS2**, **TOBJ** or **TINDEX** statement.
2. The continuation data is optional.
3. If LID1 = ALL, LB1 and UB1 will apply to all applicable load cases. In this case, no subsequent LIDs and LB, UB should be specified.
4. If a constraint references a MASSFR or INERTIA response, the load case identification number (LID) must be 0 or blank.
5. If a constraint references TRESP2 or TRESP3 responses that use the TRESP1L keyword, then the loadcase identification number (LID) must be blank.
6. Data for LID3, LID4, etc. may be specified on a continuation.
7. If a constraint references a TRESP2 or TRESP3 entry, which does not reference any TRESP1 entries, the load case identification number must be blank.
8. If LBi and UBi are specified for only some of the load cases, the response is constrained only in these specified load cases.

TCONS

Topology Design Model Data

9. Lower bounds $< -1.0\text{E}+29$ and upper bounds $> 1.0\text{E}+29$ are ignored and will not generate constraints. If LBi or UBi are blank, then no constraint will be generated for that bound.
10. If a constraint points to a FREQ response, the bounds imposed must be in Hz and only the lower or upper bound should be specified. To generate both a lower and upper bound frequency constraint, use two TRESP1 and two TCONS statements.

9.2.7 TCONS2

Data Entry: **TCONS2** - Topology Constraints.

Description: Define topology constraints using scale factors of responses' initial analysis values.

Format:

1	2	3	4	5	6	7	8	9	10
TCONS2	RID	LID1	LBF1	UBF1	LID2	LBF2	UBF2		
+		LID3	LB3F	UBF3	-etc.-				

Example:

1	2	3	4	5	6	7	8	9	10
TCONS2	16	10	0.5	1.0					

Field Information Description

2	RID	TRESP1 , TRESP2 or TRESP3 identification number (Integer > 0).
3,6	LIDi	Load case identification number (Integer ≥ 0 or ALL or blank. Default = ALL).
4,7	LBFi	Constraint lower bound imposed on this response quantity. (Real or blank. Default = no constraint).
5,8	UBFi	Constraint upper bound imposed on this response quantity. (Real or blank, $LBF \leq UBF$. Default = no constraint).

Remarks:

1. A TRESP1, TRESP2 or TRESP3 entry can be referred to by at most one TCONS2 entry. A TCONS2 entry cannot reference a TRESP1, TRESP2 or TRESP3 that is referred to by a **TCONS**, **TOBJ** or **TINDEX** statement.
2. The continuation data is optional.
3. If LID1 = ALL, LBF1 and UBF1 will apply to all applicable load cases. In this case, no subsequent LIDs and LB, UB should be specified. However, the actual bound that will be used in each loadcase will not necessary be the same.
4. If a constraint references a MASSFR or INERTIA response, the load case identification number (LID) must be 0 or blank.
5. If a constraint references TRESP2 or TRESP3 responses that use the TRESP1L keyword, then the loadcase identification number (LID) must be blank.
6. Data for LID3, LID4, etc. may be specified on a continuation.

7. If a constraint references a TRESP2 or TRESP3 entry, which does not reference any TRESP1 entries, the load case identification number must be blank.
8. If LBFi and UBFi are specified for only some of the load cases, the response is constrained only in these specified load cases.
9. If LBFi or UBFi are blank, then no constraint will be generated for that bound.
10. If a constraint points to a FREQ response, only the lower or upper bound should be specified. To generate both a lower and upper bound frequency constraint, use two TRESP1 and two TCONS2 statements.
11. Additional information on TCONS2 can be found in the following chapter:
Relative Constraint Bounds (p. 328)
12. Scaling a negative number with a factor above 1.0 will result in a number less than that with a factor below 1.0. Therefore, care should be taken when using responses that may have negative values to insure that the calculated lower bound is less than the calculated upper bound. If you wish to set a scale factor bound on the magnitude of such a response, then it is recommended to use a TRESP3 with the NORM1 function.

9.2.8 TCYCLE

Data Entry: **TCYCLE** - Topology Progressive Rule Definition.

Description: Define an optional progressive rule schedule .

Format:

1	2	3	4	5	6	7	8	9	10
TCYCLE									
+	DESMIN1	DESMAX1	RV1						
+	DESMIN2	DESMAX2	RV2						
+	-etc.-								

Example:

1	2	3	4	5	6	7	8	9	10
TINDEX									
+	2	8	1.80						
+	2	8	2.40						
+	2	8	3.00						

Field Information Description

2	DESMIN _i	Minimum number of design cycles for stage i.
3	DESMAX _i	Maximum number of design cycles for stage i.
4	RV1 _i	Power rule value

Remarks:

1. Only one TCYCLE statement is allowed in the input data.
2. The DOPT parameter **TCYCLEM** must be set to 1 for the progressive rule in this table to be used. A value of 0 for TCYCLEM (the default) will cause this table to be ignored.
3. This data is optional. If the DOPT parameter TCYCLEM is set to 1 and no TCYCLE entry is present in the input data, the program will use a default progressive rule.
4. If the DOPT parameter **DESMAXM** is set to 1, then the sum of all DESMAX_i in the TCYCLE table will override the DESMAX value in DOPT. If the DOPT parameter DESMAXM is set to 0, then DESMAX in DOPT will be used, in which case this schedule might not be completed or the number of allowed design cycle might exceed the values in the table. If the number of design cycles exceeds the values in the table, the program will use the RV1_i value of the last stage for all extra design cycles.

5. The progressive rule can be used with all types of topology methods available in the program (element based, geometry based and hybrid based).
6. See [Progressive Rule](#) (Sec. 5.2.17) for additional information.

9.2.9 TINDEX

Data Entry: **TINDEX** - Topology Compliance Index Definition.

Description: Define the topology compliance index objective function.

Format:

1	2	3	4	5	6	7	8	9	10
TINDEX	RID1	LID1	W1	RID2	LID2	W2	RID3	LID3	W3
+	RID4	LID4	W4	-etc.-					

Example:

1	2	3	4	5	6	7	8	9	10
TINDEX	1	101	0.33						
+	2	102	0.33						
+	3	103	0.33						

Field Information Description

2, 5, 8	RIDi	Response entry (TRESP1 , TRESP2 or TRESP3) ID. (Integer > 0).
3, 6, 9	LIDi	Load case identification number. (Integer > 0 or blank, Default = the first load case) If RID corresponds to MASSFR response, LID is ignored.
4	W1	Weighting factor. (Real \neq 0). See Remark 4.
7, 10, 4, ...	Wi	Weighting factor. (Real \neq 0 or one of the words "MULT" or "DIV"). See Remarks 4 and 5.

Remarks:

1. Only one TINDEX statement is allowed in the input data.
2. TRESP1, TRESP2 and TRESP3 entries referenced by the TINDEX data can define only a single response per load case.
3. Only one of **TOBJ** or TINDEX is allowed in the input data.
4. If TINDEX references TRESP2 and/or TRESP3 responses that use the TRESP1L keyword, then the corresponding loadcase ID (LID) must be the same as the first LID specified in the TRESP1L data.

- If the DOPT parameter, TINDEXM, is 0 or 1, then responses are normalized by their values in the first design cycle. If TINDEXM is 2 or 3, then responses are not normalized.

- TINDEXM = 0 or 1

$$R_i = \frac{Resp_i}{|Resp_{0i}|}$$

- TINDEXM = 2 or 3

$$R_i = Resp_i$$

- The compliance index objective function is calculated using the following equation:

$$T = \sum_{i=1} f_i$$

If the DOPT parameter TINDEXM is 0 or 2, then the compliance index objective function terms are calculated using reciprocals for negative weighting factors. If TINDEXM is 1 or 3, then the compliance index objective function is a straight summation.

- TINDEXM = 0 or 2

$$f_i = \begin{cases} W_i \cdot R_i & \text{if } W_i > 0 \\ \frac{-W_i}{R_i} & \text{if } W_i < 0 \end{cases}$$

- TINDEXM = 1 or 3

$$f_i = W_i R_i$$

Note that in all cases, using a positive weighting factor will drive the corresponding response to be minimized, while a negative weighting factor will drive the corresponding response to be maximized.

- If the weighting factor, W_i , is the word MULT or DIV, then the corresponding response, R_i , does not make a new term f_i in the compliance index function. Instead, R_i multiplies or divides the immediately previous term f_j ($j < i$) in the compliance index function for which a real weighting factor was given. For example:

	1	2	3	4	5	6	7	8	9	10
TINDEX	1	101	1.0							
+	2	102	MULT							
+	3	103	2.0							
+	4	104	DIV							

For the above entry, the objective function is $T = R_1 R_2 + 2.0 \frac{R_3}{R_4}$

8. Recommended Weighting factor signs are;

Response	W_i Sign
FREQ	NEGATIVE
MASSFR SENERGY	POSITIVE

9. The compliance index objective function is always minimized.

9.2.10 TOBJ

Data Entry: **TOBJ** - Design Objective.

Description: Define topology objective function.

Format:

1	2	3	4	5	6	7	8	9	10
TOBJ	RID	LABEL	LID	MIN/MAX					

Example:

1	2	3	4	5	6	7	8	9	10
TOBJ	14	TOPIND		MIN					

Field	Information	Description
2	RID	Response entry (TRESP1 , TRESP2 or TRESP3) ID. This identifies the response that is to be the objective function. (Integer > 0).
3	LABEL	User defined name for output purposes (Characters or blank).
4	LID	Load case identification number. (Integer > 0 or blank, Default = the first load case) If RID corresponds to massfr response, LID is ignored.
5	MIN/MAX	Defines minimization (MIN) or maximization (MAX) to be performed. (Default = MIN).

Remarks:

1. Only one TOBJ statement is allowed in the input data.
2. TOBJ and **TINDEX** data cannot be used in the same input data.
3. The TRESP1, TRESP2 or TRESP3 entry referenced by the TOBJ data can define only a single response per load case.
4. If the objective function references a TRESP2 or TRESP3 response that uses the TRESP1L keyword, then the loadcase ID (LID) must be the same as the first LID specified in the TRESP1L data.

9.2.11 TPROP

Data Entry: **TPROP** - Designable Region Selection.

Description: Define a topological designable region.

Format:

1	2	3	4	5	6	7	8	9	10
TPROP	ID	PTYPE	PID	INIT	TMIN	DELT	DTMIN	TSYMID	
+	"RULE"	RTYPE	RV1	RV2	RV3				
+	"PLINK"	PID1	PID2	PID3	PID4	PID5	PID6	PID7	PID8
+		PID9	PID10	-etc.-					

Legacy Format:

1	2	3	4	5	6	7	8	9	10
TPROP	ID	PID	RTYPE	INIT	TMIN	DELT	DTMIN	TSYMID	
+	RV1	RV2	RV3						

Example:

1	2	3	4	5	6	7	8	9	10
TPROP	12	PSHELL	25	0.3					

Field Information Description

2	ID	Unique topological region ID. (Integer > 0).
4	PTYPE	Designable property type. One of the following words: PROD, PBAR, PBARL, PBEAM, PBEAML, PCOMP, PCOMPG, PSHELL, PSOLID, PSHEAR, PWELD, PAXIS, PROP or blank (Default = PROP).
3	PID	Property identification number. (Integer > 0).
5	INIT	Initial mass fraction value. ($0.0 \leq \text{Real} \leq 1.0$). See Remark 6.
6	TMIN	Minimum mass fraction. $0.0 \leq \text{Real} < 1.0$ or blank (Default = 0.0). See Remark 1.
7	DELT	Fractional move limit. Real > 0.0 or blank. (Default = 10^{-6}). See Remark 1.
8	DTMIN	Minimum move limit. Real > 0.0 or blank. (Default = 0.20). See Remark 1.
9	TSYMID	TSYM identification number for fabrication constraints. (Integer or blank).

2	RULE	Word indicating that the following data correspond to mixing rule data. This continuation is optional. Defaults will be used if this line is omitted. See Remarks 2 and 3
3	RTYPE	Mixing rule: POWER or MIX. (Default = POWER). See Remarks 2 and 3
4, 5, 6	RV1, RV2, RV3	Mixing rule factors. (Real or blank). See Remarks 2 and 3
2	PLINK	Word indicating that the following data correspond to property identification numbers. This data is optional. See Remark 7.
3	PID1	Property identification number (Integer>0). See Remark 7.
4,...	PID2...PIDn	Property identification numbers (Integer>0 or Blank). See Remark 7.

Remarks:

1. The default values for TMIN, DELT and DTMIN can be changed using the DOPT data statement.
2. When RTYPE = POWER, the following relationships are used:

$$E = E_{\text{MIN}} + (E_0 - E_{\text{MIN}})X^{\text{RV1}}$$

$$\rho = \rho_0 X$$

where $E_{\text{MIN}} = E_0 \cdot \text{RV2}$. In this case, RV3 is ignored. (Default: RV1 = 3.0, RV2 = 10^{-6})

3. When RTYPE = MIX, the following relationships are used:

$$E = E_0 \left\{ \text{RV1} [X + \text{RV2}(1 - X)] + (1 - \text{RV1}) \frac{\text{RV2}}{\text{RV2} \cdot X + 1 - X} \right\}$$

$$\rho = \rho_0 [X + \text{RV3}(1 - X)]$$

Note:

If RV1 = 0.0, then $E = E_0 \left\{ \frac{\text{RV2}}{\text{RV2} \cdot X + 1 - X} \right\}$. This corresponds to the Reuss

(Isostress) mixing rule.

If RV1 = 1.0, then $E = E_0 [X + \text{RV2}(1 - X)]$. This corresponds to the Voigt

(Isostrain) mixing rule. (Default: RV1 = 0.0, RV2 = 10^{-6} , RV3 = 10^{-6})

4. PSHELL properties referenced in TPROP have to reference MAT1 materials. Otherwise, a fatal error message will be issued by GENESIS.
5. If PTYPE is PROP or blank, the identification number PID must be uniquely defined among all property entries, and the property must be one of the topologically designable types.

6. The initial value, INIT, in field 5 is ignored for topology regions with stamping fabrication constraints with the “NO” punch hole option. For this case a value of 1.0 is used.
7. The optional PLINK list is used to create an “extended topology region”. Extended topology regions are primarily used to enforce fabrication constraints across different properties. Property PID is called the master property. The properties that are listed in the PLINK list are called slave properties. Properties can be listed on at most one entry (either as PID or in a PLINK list). Slave properties have to be of the same type as the master property (e.g., a PBAR master can not have a PBEAM as a slave property).

Topologically Designable Elements.

Elements	Properties	Material
CROD	PROD	MAT1
CBAR	PBAR PBARL	MAT1
CBEAM	PBEAM PBEAML	MAT1
CWELD	PWELD	MAT1
CTRIA3 CQUAD4 CTRIA6 CQUAD8	PSHELL	MAT1
CTRIA3 CQUAD4 CTRIA6 CQUAD8	PCOMP PCOMPG	MAT1 / MAT8
CSHEAR	PSHEAR	MAT1
CTRIAX6	PAXIS	MAT1
CTETRA CPENTA CHEXA CHEX20 CPYRA	PSOLID	MAT1, MAT9 or MAT11

9.2.12 TPROPC

Data Entry: **TPROPC** - Clone Topology Designable Region Selection.

Description: Define a clone (copy) topological designable region.

Format:

1	2	3	4	5	6	7	8	9	10
TPROPC	ID	PTYPEC	PIDC	CIDC	PTYPEP	PIDP	CIDP		
+	SCALEX	SCALEY	SCALEZ						

Example:

1	2	3	4	5	6	7	8	9	10
TPROPC	12	PSHELL	101	2	PSHELL	100	1		

Field Information Description

2	ID	Unique topological region ID. (Integer > 0).
3	PTYPEC	Designable clone property type. One of the following words: PROD, PBAR, PBARL, PBEAM, PBEAML, PCOMP, PCOMPG, PSHELL, PSOLID, PSHEAR, PWELD, PAXIS, PROP or blank (Default = PROP).
4	PIDC	Property identification number of clone property. (Integer > 0).
5	CIDC	Reference coordinate system identification number of the clone property. Integer ≥ 0 or Blank (Default = 0).
6	PTYPEP	Designable parent property type. One of the following words: PROD, PBAR, PBARL, PBEAM, PBEAML, PCOMP, PCOMPG, PSHELL, PSOLID, PSHEAR, PWELD, PAXIS, PROP or blank (Default = PROP).
7	PIDP	Property identification number of parent property. (Integer > 0).
8	CIDP	Reference coordinate system identification number of the parent property. Integer ≥ 0 or Blank (Default = 0).
2	SCALEX	Scale factor along the x-axis of CIDC/CIDP. Real $\neq 0$ or blank (Default = 1.0). See Remarks 5.
3	SCALEY	Scale factor along the y-axis of CIDC/CIDP. Real $\neq 0$ or blank (Default = 1.0). See Remarks 5.
4	SCALEZ	Scale factor along the z-axis of CIDC/CIDP. Real $\neq 0$ or blank (Default = 1.0). See Remarks 5.

Remarks:

1. The parent property must be listed on a **TPROP** data entry.

2. A cloned property can not be listed on **TPROP** or on another TPROPC data entry.
3. A parent property can be referenced by multiple TPROPC data entries.
4. A coordinate system ID of zero references the basic coordinate system.
5. A scale factor magnitude greater than 1.0 will cause the topology design in the clone property to be larger than in the parent property. Conversely, a scale factor magnitude less than 1.0 will cause the topology design in the clone property to be smaller than in the parent property. A scale factor magnitude of 1.0 produces an exact copy in the clone. A negative scale factor will cause the design to be reversed along the corresponding axis.
6. When viewed with respect to coordinate system CIDC, the cloned property topology design will match with that in the parent property viewed with respect to CIDP, subject to scaling along each of the coordinate axes as discussed in remark 5. Except for rare special cases, CIDC and CIDP should be different.
7. If PTYPEC or PTYPEP is PROP or blank, the corresponding identification number PIDC or PIDP must be uniquely defined among all property entries, and the property must be one of the topologically designable types.
8. When minimum member size is used (defined in **TSYM1**, **TSYM2** or **TSYM3** that is reference in the **TPROP** associated to the parent), the clone mesh does not need to be identical to the parent mesh. However, if minimum member size is not used, the clone mesh needs to match its parent mesh to be cloned.
9. When non-unit scale factor(s) are used and minimum member size is not used, the clone mesh must be a scaled version of the parent mesh. It is recommended to set a minimum member size on the parent when using non-unit scale factors.
10. Additional information on TPROPC can be found in the following chapter: **Cloned Topology Regions** (Sec. 5.2.7)

9.2.13 TRESP1

Data Entry: **TRESP1** - Select Fundamental Response Quantities.

Description: Define a set of structural responses that are used in topology design, either as constraint or as an objective.

Format:

1	2	3	4	5	6	7	8	9	10
TRESP1	ID	LABEL	RTYPE	PTYPE		ATTA	ATT1	ATT2	ATT3
+	ATT4	-etc.-							

Example:

1	2	3	4	5	6	7	8	9	10
TRESP1	10	OBJ	SENERGY						

Field Information Description

2	ID	Unique identification number. (Integer > 0).
3	LABEL	User defined label (or blank).
4	RTYPE	Response type. One of DISP, RELDISP, SPCF, FREQ, MASSFR, SENERGY, VMINDEX, CPRESS, TEMP, HTC, LAMA, DDISP, DVELO, DACCE, DDISPS, DVELO, DACCES, MDISP, MVELO, MACCE, MDISPS, MVELO, MACCES, PSDDISP, PSDVELO, PSDACCE, PSDDDS, PSDVS, PSDAS, RMSDISP, RMSVELO, RMSACCE, UFDISP, UFVELO, UFACCE, UFDISPS, UFVELO, UFACCES, ERP, ERPS, or INERTIA. See table below.
5	PTYPE	If RTYPE is MASSFR, then PTYPE = PROP, If RTYPE=FREQ or LAMA then PTYPE is mode number, otherwise PTYPE is blank.
7	ATTA	Displacement, velocity, acceleration, grid contact pressure, or inertia component number, or blank if RTYPE is MASSFR, VMINDEX, FREQ or LAMA. See table below.,
8	ATT1	Grid point if RTYPE = DISP, RELDISP, SPCF, CPRESS, TEMP, DDISP, DVELO, DACCE, MDISP, MVELO, MACCE, PSDDISP, PSDVELO, PSDACCE, RMSDISP, RMSVELO or RMSACCE. Property ID if RTYPE = MASSFR. DSHIFT ID if RTYPE =DDISPS, DVELO, DACCES, MDISPS, MVELO, MACCES, PSDDDS, PSDVS, PSDAS, UFDISPS, UFVELO, UFACCES or ERPS.

9	ATT2	Grid, Spoint, Property or Field point ID (Integer > 0 or blank). Grid/Spoint ID if RTYPE = DISP, RELDISP, SPCF, CPRESS, TEMP, DDISP, DVELO, DACCE, DDISPS, DVELO, DACCES, MDISP, MVELO, MACCE, MDISPS, MVELO, MACCES, PSDDISP, PSDVELO, PSDACCE, PSDDS, PSDVS, PSDAS, RMSDISP, RMSVELO or RMSACCE. Property ID if RTYPE = MASSFR. Field point ID if RTYPE = UFDISP, UFVELO or UFACCE, UFDISPS, UFVELO or UFACCES Element set ID in ERPPNL if RTYPE=ERP or ERPS The keyword "THRU" can be used with static displacement. See Remark 4.
10,2,3,...	ATTi	Grid or Spoint ID numbers, (Integer > 0 or blank). See table below.

RESPONSE	RTYPE	PTYPE	ATTA	ATTi
Static Displacement	DISP	Blank	Displacement component code (1-7).	Grid or Spoint ID, "THRU" or "ALL"
Static Relative Displacement	RELDISP	Blank	Displacement component code (1-7).	Grid or Spoint ID
Reaction Force	SPCF	Blank	Reaction force component code (1-7).	Grid or Spoint ID
Grid Contact Pressure	CPRESS	Blank	Cpress component code (1).	Grid ID, "THRU" or "ALL"
Natural Frequency	FREQ	Mode number	Blank	Blank
Mass Fraction	MASSFR	Blank or PROP	Blank	Blank or Property ID
Static Strain Energy	SENERGY	Blank or PROP or PAXIS or PBAR or PBARL or PBEAM or PBEAML or PBUSH or PROD or PSHELL or PSHEAR or PCOMP or PCOMPG or PSOLID or PELAS or PVECTOR or PBUSH or PBUSHT or PGARP or PWELD or PK2UU	Blank	Blank or Property ID
Von Mises Stress Index	VMINDEX	Blank	Blank	Blank
Temperature	TEMP	Blank	Blank	Grid or Spoint ID, "THRU" or "ALL"
Heat Transfer Compliance	HTC	Blank	Blank	Blank
System Inertia Matrix Component	INERTIA	Blank	Inertia item code (1-22)	Blank
Buckling Load Factor	LAMA	Mode number	Blank	Blank

Dynamic Displacement, Velocity, Acceleration (From Direct Loadcases)	DDISP, DVELO, DACCE	Blank	Component code: (1-24)	Grid or Spoint ID
Shifted Dynamic Displacement, Velocity, Acceleration (From Direct Loadcases)	DDISPS, DVELO, DACCES	Blank	Component code: (1-6)	ATT1: DSHIFT ID. ATTi (i>1): Grid or Spoint ID
Dynamic Displacement, Velocity, Acceleration (From Modal Loadcases)	MDISP, MVELO, MACCE	Blank	Component code: (1-24)	Grid or Spoint ID
Shifted Dynamic Displacement, Velocity, Acceleration (From Modal Loadcases)	MDISPS, MVELO, MACCES	Blank	Component code: (1-6)	ATT1: DSHIFT ID. ATTi (i>1): Grid or Spoint ID
Random Power Spectral Density Dynamic Displacement, Velocity, Acceleration (From Direct or Modal Loadcases)	PSDDISP, PSDVELO, PSDACCE	Blank	Component code: (1-24)	Grid or Spoint ID
Shifted Random Power Spectral Density Dynamic Displacement, Velocity, Acceleration (From Direct or Modal Loadcases)	PSDDDS, PSDVS, PSDAS	Blank	Component code: (1-6)	ATT1: DSHIFT ID. ATTi (i>1): Grid or Spoint ID

Random Dynamic Displacement, Velocity, Acceleration (From Direct or Modal Loadcases)	RMSDISP, RMSVELO, RMSACCE	Blank	Component code (1-6)	Grid or Spoint ID
User Function of Dynamic Displacement, Velocity, Acceleration	UFDISP, UFVELO, UFACCE	Blank	Field Point item code 1: Magnitude 2: Phase 3: Real 4: Imaginary	Field Point ID (Defined in UFDATA file)
Shifted User Function of Dynamic Displacement, Velocity, Acceleration (From Direct or Modal Loadcases)	UFDISPS, UFVELOS, UFACCES	Blank	Component code 1: Magnitude	ATT1: DSHIFT ID. ATTi (i>1):Field Point ID (Defined in UFDATA file)
Equivalent Radiated Power	ERP	Blank	Item code 1: Actual ERP value 2: Decibel scale value	Element set ID (Specified in ERPPNL entry)
Shifted Equivalent Radiated Power	ERPS	Blank	Item code 1: Actual ERP value 2: Decibel scale value	ATT1: DSHIFT ID. ATTi (i>1): Element set ID (Specified in ERPPNL entry)

Remarks:

1. TRESP1 entries must have unique identification numbers with respect to **TRESP2** and **TRESP3** entries.
2. If RTYPE=MASSFR, the response is the mass fraction, which is the mass of designed elements (all or of one property) divided by the mass calculated when all design variables are 1.0.
3. The displacement component code should be 1 for SPOINT.

4. If RTYPE=RELDISP, the number of ATTi items must be even. Each pair [ATT(2k-1), ATT(2k)] makes a response that is the difference: the displacement at the second grid of the pair minus the displacement at the first grid of the pair.

	1	2	3	4	5	6	7	8	9	10
TRESP1	5	RELAT	RELDISP				2	1001	1002	23
+	24									

5. The keyword “THRU” can be used with displacement to specify the grid points. For example:

	1	2	3	4	5	6	7	8	9	10
TRESP1	7	PANEL	DISP				2	1	THRU	100
+	200	300	THRU	320						

specifies grid points 1, 2, 3, . . . , 100, 200, 300, 301, 302, . . . , 320.

6. The keyword “ALL” can be used with displacement to specify all of the grid points. For example:

	1	2	3	4	5	6	7	8	9	10
TRESP1	10	MESH	DISP					ALL		

7. If RTYPE = DDISP, DVELO, DACCE, MDISP, MVELO or MACCE, then only one component can be specified. The components for grid points are 1-6 for magnitude, 7-12 for phase, 13-18 for real and 19-24 for imaginary.
8. If RTYPE = PSDDISP, PSDVELO, PSDACCE, PSDDS, PSDVS or PSDAS, then only one component can be specified. The components for grid points are 1-6. The component code for scalar points for dynamic responses should be 1.
9. If RTYPE = RMSDISP, RMSVELO or RMSACCE, then only one component can be specified. The components for grid points are 1-6.
10. If RTYPE = UFDISP, UFVELO or UFACCE, the user function is defined by file specified by the UFDATA executive control command. The field point item codes are: 1 for magnitude, 2 for phase, 3 for real and 4 for imaginary.
11. If RTYPE = UFDISPS, UFVELOS, or UFACCES, the user function is defined by file specified by the UFDATA executive control command. The field point item codes should be 1.
12. For shifted responses (RTYPE=DDISPS, DVELOS, DACCES, MDISPS, MVELOS, MACCES, PSDDS, PSDVS, PSDAS, UFDISP, UFVELOS, UFACCES or ERPS) the **DSHIFT** data entry is required.
13. If RTYPE = CPRESS, the response is the grid contact (or glue connection) pressure measured at the grid in the normal direction of the contact surface (ATTA=1). The contact pressure is calculated by dividing the normal contact force at the grid by the associated area.

14. If RTYPE = ERP or ERPS, the element set id must also be specified on an ERPPNL entry. The item code should be either 1 (for linear scale) or 2 (for decibel scale). For RTYPE=ERPS, if item code is 2, then the corresponding data can only use the FTYPE=IDENT option (LOG or LOG10 option in can not be applied to ERP in decibel scale).
15. If RTYPE = HTC, the response is the heat transfer compliance, which is defined as
$$HTC = \frac{1}{2} \{T\}^T \{F\}$$
, where $\{T\}$ is the vector of grid temperatures, and $\{F\}$ is the vector of applied heat fluxes.
16. If RTYPE = VMINDEX, the response is the von Mises stress index, which is a global value that estimates the maximum von Mises stress in elements. The von Mises stress limit must be defined in the SS field of MAT1 entries. Elements that reference MAT1 entries that do not define SS are not included in the index. If the response value is greater than 1.0, then either some elements have a von Mises stress significantly higher than SS, or there are a significant number of elements with von Mises stress somewhat higher than SS. Note that because the index is merely an estimate, having the response value less than 1.0 does not guarantee that all elements have von Mises stress less than SS, but the number of violating elements should be a small fraction of the total number of elements. The VMINDEX estimate is most accurate for solid elements, and significantly less accurate for 1-D and 2-D elements.
17. Displacements, velocites, accelerations and inertia item codes are listed below.

DISPLACEMENT ITEM CODES

Item #	Meaning
1	Ux: Displacement along the x direction in the output coordinate system
2	Uy: Displacement along the y direction in the output coordinate system
3	Uz :Displacement along the z direction in the output coordinate system
4	Θ_x : Rotation with respect the x axis in the output coordinate system
5	Θ_y : Rotation with respect the y axis in the output coordinate system
6	Θ_z : :Rotation with respect the z axis in the output coordinate system
7	Spherical (Total) displacement = $\text{SQRT}(U_x^{**2} + U_y^{**2} + U_z^{**2})$

REACTION FORCE (SPCF) ITEM CODES

Item #	Meaning
1	RFx: Reaction Force along the x direction in the output coordinate system
2	RFy: Reaction Force along the y direction in the output coordinate system
3	RFz :Reaction Force along the z direction in the output coordinate system
4	RMx: Reaction Force along the x direction in the output coordinate system
5	RMx: Reaction Force along the x direction in the output coordinate system
6	RMz: Reaction Force along the x direction in the output coordinate system
7	Spherical (Total) Reaction Force = $\text{SQRT}(\text{RFx}^2 + \text{RFy}^2 + \text{RFz}^2)$

CPRESS ITEM CODE
(Nonlinear Contact Analysis Only)

Item #				
Static				
1				Grid contact pressure - normal to the contact surface or Grid glue connection pressure - normal to the connecting surfaces

DDISP, MDISP, DVELO, MVELO, DACCE & MACCE ITEM CODES

User Item #				Stress
Magnitude	Phase	Real Comp.	Imaginary Comp.	
1	7	13	19	Displacement, velocity or acceleration along the x direction in the output coordinate system
2	8	14	20	Displacement, velocity or acceleration along the y direction in the output coordinate system
3	9	15	21	Displacement, velocity or acceleration along the z direction in the output coordinate system
4	10	16	22	Displacement, velocity or acceleration rotation with respect the x axis in the output coordinate system
5	11	17	23	Displacement, velocity or acceleration rotation with respect the y axis in the output coordinate system
6	12	18	24	Displacement, velocity or acceleration rotation with respect the z axis in the output coordinate system

Equivalent Radiated Power ITEM CODE

Item #	Meaning
1	ERP - Equivalent Radiated Power in Standard Scale $ERP = \frac{1}{2} \rho c \iint v_n^* v_n dA$
2	Equivalent Radiated Power in Standard in Decibel Scale $ERP(dB) = 10 \log \left(\frac{RHOCP}{ERPREFDB} ERP \right)$

SYSTEM INERTIA ITEM CODES

Item #	Meaning
1	Ixx at center of gravity
2	Iyy at center of gravity
3	Izz at center of gravity
4	Ixy at center of gravity
5	Iyz at center of gravity
6	Izx at center of gravity
7	Principal 1 at center of gravity
8	Principal 2 at center of gravity
9	Principal 3 at center of gravity
10	Ixx at grdpnt
11	Iyy at grdpnt
12	Izz at grdpnt
13	Ixy at grdpnt
14	Iyz at grdpnt
15	Izx at grdpnt
16	Principal 1 at grdpnt
17	Principal 2 at grdpnt
18	Principal 3 at grdpnt
20	Y center of gravity with respect to grdpnt
21	Z center of gravity with respect to grdpnt
22	X center of gravity with respect to grdpnt

9.2.14 TRESP2

Data Entry: **TRESP2** - Create Equation Response Quantities.

Description: Define equation responses that are used in the design, either as an objective function or as constraints.

Format:

1	2	3	4	5	6	7	8	9	10
TRESP2	ID	LABEL	EQID	REGION					
+	"TVAR"	NDV1	NDV2	NDV3	NDV4	NDV5	NDV6	NDV7	NDV8
+		NDV9	-etc.-						
+	"DTABLE"	NC1	NC2	NC3	NC4	NC5	NC6	NC7	NC8
+		NC9	-etc.-						
+	"TRESP1"	NR1	NR2	NR3	NR4	NR5	NR6	NR7	NR8
+		NR9	-etc.-						

Alternate Format 1:

1	2	3	4	5	6	7	8	9	10
TRESP2	ID	LABEL	EQID						
+	"TVAR"	NDV1	NDV2	NDV3	NDV4	NDV5	NDV6	NDV7	NDV8
+		NDV9	-etc.-						
+	"DTABLE"	NC1	NC2	NC3	NC4	NC5	NC6	NC7	NC8
+		NC9	-etc.-						
+	"TRESP1L"	NR1	LIDR1	NR2	LIDR2	NR3	LIDR3	NR4	LIDR4
+		NR5	LIDR5	-etc.-					

Alternate Format 2:

1	2	3	4	5	6	7	8	9	10
TRESP2	ID	LABEL	EQID						
+	"TVAR"	NDV1	NDV2	NDV3	NDV4	NDV5	NDV6	NDV7	NDV8
+		NDV9	-etc.-						
+	"DTABLE"	NC1	NC2	NC3	NC4	NC5	NC6	NC7	NC8
+		NC9	-etc.-						
+	"TRESP2"	NS1	NS2	NS3	NS4	NS5	NS6	NS7	NS8
+		NS9	-etc.-						

Example:

	1	2	3	4	5	6	7	8	9	10
TRESP2	1	EBUCK	3	4						
+	TVAR	87	25	3	8					
+	DTABLE	VAL1	YOUNGS	CVAL4						
+	TRESP1	12	5	2	102	202	302	402	502	
+		602	702							

Field Information Description

2	ID	Unique identification number. (Integer > 0). See remark 6.
3	LABEL	User defined label for information only (Characters or blank).
4	EQID	DEQATN entry identification number. (Integer > 0).
5	REGION	Region identifier for constraint screening. (Integer > 0 or blank. Default puts this set of responses in a unique region.).
2	TVAR	Word indicating TVAR identification numbers.
3,4,..	NDVi	TVAR identification number. (Integer > 0 or Blank or "THRU").
2	DTABLE	Word indicating constant identification numbers in DTABLE.
3,4,..	NCj	Name of constant defined in DTABLE input. (Character). See Remark 17.
2	TRESP1	Word indicating TRESP1 identification numbers.
3,4,..	NRk	TRESP1 identification number. (Integer > 0 or Blank or "THRU"). See remarks 2, 4 and 5.
2	TRESP1L	Word indicating TRESP1 and Loadcase identification numbers.
3,5,..	NRk	TRESP1 identification number (Integer > 0 or Blank). See remark 12.
4,6,..	LIDRk	Loadcase identification number (Integer > 0).
2	TRESP2	Word indicating TRESP2 or TRESP3 identification numbers.
3,4,..	NSk	TRESP2 or TRESP3 identification number. (Integer > 0 or Blank or "THRU").

Remarks:

1. TRESP2 entries can reference TVAR, DTABLE, TRESP1, TRESP2 and TRESP3 entries.

2. If the TRESP1 keyword is used, then the TRESP1 responses referenced by the TRESP2 entry must all be of the same analysis type (static, frequency, direct dynamic, modal dynamic, random, heat transfer or buckling). Static, frequency, heat transfer or buckling responses cannot be mixed. Mass and volume responses cannot be mixed with static, frequency, direct dynamic, modal dynamic, heat transfer or buckling responses.
3. Different static responses **can** be mixed on a single TRESP2 entry. For example, strain energy and displacement responses can both be referenced by a single TRESP2 entry.
4. TRESP2 can refer to TRESP1's that have multiple physical responses associated with them. However, each referenced TRESP1 has to have the same number of responses. For PTYPE = PROP, the number of responses is the sum of the elements associated with each property entered in the referenced TRESP1.
5. For RTYPE = DISP, the number of ATTi may be different on each TRESP1 entry, but the number of ATTi multiplied by the number of components must be the same. DISP responses on the TRESP1 entries can reference different grids and components (directions).
6. TRESP2 entries must have unique identification numbers with respect to **TRESP1** and **TRESP3** entries.
7. The sequence in which TVAR, DTABLE and TRESP1 / TRESP1L / TRESP2 occur is not arbitrary. They must be in the order shown. Any of these five words along with the identification numbers associated with them can be omitted if they are not involved in this TRESP2 relationship. However, at least one of the words TVAR, TRESP1 / TRESP1L / TRESP2 should exist.
8. TRESP2 entries that reference TRESP1 entries which are mass fraction responses will be in a distinct REGION. Two different TRESP2 entries that reference static, frequency, direct dynamic, modal dynamic, heat transfer and/or buckling responses will never be in the same REGION, even if they are assigned the same REGION identification number. TRESP1 and TRESP2 responses will never be contained in the same region, even if they are assigned the same REGION identification number. The default is to put all responses referenced by one TRESP2 entry in a distinct region.
9. If a TRESP2 entry is not referenced by a TOBJ, TCONS or TINDEX entry, this response is calculated for all load cases and printed with other TRESP2 values in the TRESP2/3 response table.
10. The variables identified by NDVi, NCj, NRk and NSk correspond to parameters listed in the left-hand side of the equation on the DEQATN entry identified by EQID. The parameters are assumed to be; first, according to the order of TVAR, DTABLE and TRESP1 / TRESP2 specifications, and second, NDVi, NCj and NRk / NSk. In the example above, the order is: TVAR 87, 25, etc.; then DTABLE 1, 8, etc.; and finally TRESP1 12, 5, etc.
11. If the TRESP1L keyword is used, the responses from different analysis types; static, frequency, direct dynamic, modal dynamic, heat transfer, and/or buckling can be mixed.
12. If the TRESP1L keyword is used, then the LID on the TCONS data must be blank.

13. If the TRESP1L keyword is used, then the LID on the TOBJ data must be the same as the LID of the first TRESP1 referenced (LIDR1).
14. If the TRESP1L keyword is used and the response type is MASSFR or INERTIA, then the LID should be the ID of the first LOADCASE.
15. If the TRESP1L keyword is used with dynamic responses, then each dynamic loadcase referenced (LIDi) must have the same number of loading frequencies.
16. The name CYCLE can be used in the DTABLE list to pass the current value of the design cycle number (as long as the name CYCLE is not explicitly defined on the DTABLE entry).
17. NDVi, NRk and NSk can be an integer or a string “THRU”. If a NDVi is “THRU”, it specifies a sequential list of TVAR; if a NRk is “THRU”, it specifies a sequential list of TRESP1; if a NSk is “THRU”, it specifies a sequential list of TRESP2 or TRESP3.

9.2.15 TRESP3

Data Entry: **TRESP3** - Create Synthetic Response Quantities.

Description: Define user-subroutine or built-in responses that can be used in the design, either as constraint or as an objective.

Format:

1	2	3	4	5	6	7	8	9	10
TRESP3	ID	LABEL	LIBID or NAME	REGION					
+	"TVAR"	NDV1	NDV2	NDV3	NDV4	NDV5	NDV6	NDV7	NDV8
+		NDV9	-etc.-						
+	"DTABLE"	NC1	NC2	NC3	NC4	NC5	NC6	NC7	NC8
+		NC9	-etc.-						
+	"TRESP1"	NR1	NR2	NR3	NR4	NR5	NR6	NR7	NR8
+		NR9	-etc.-						

Alternate Format 1:

1	2	3	4	5	6	7	8	9	10
TRESP3	ID	LABEL	LIBID or NAME						
+	"TVAR"	NDV1	NDV2	NDV3	NDV4	NDV5	NDV6	NDV7	NDV8
+		NDV9	-etc.-						
+	"DTABLE"	NC1	NC2	NC3	NC4	NC5	NC6	NC7	NC8
+		NC9	-etc.-						
+	"TRESP1L"	NR1	LIDR1	NR2	LIDR2	NR3	LIDR3	NR4	LIDR4
+		NR5	LIDR5	-etc.-					

Alternate Format 2:

1	2	3	4	5	6	7	8	9	10
TRESP3	ID	LABEL	LIBID or NAME						
+	"TVAR"	NDV1	NDV2	NDV3	NDV4	NDV5	NDV6	NDV7	NDV8
+		NDV9	-etc.-						
+	"DTABLE"	NC1	NC2	NC3	NC4	NC5	NC6	NC7	NC8
+		NC9	-etc.-						
+	"TRESP2"	NS1	NS2	NS3	NS4	NS5	NS6	NS7	NS8
+		NS9	-etc.-						

Example: User-Subroutine Name - GNSUB3

	1	2	3	4	5	6	7	8	9	10
TRESP3	1	EBUCK	3	4						
+	TVAR	87	25	3	8					
+	DTABLE	VAL1	YOUNGS	CVAL4						
+	TRESP1	12	5	2	102	202	302	402	502	
+		602	702							

Example: Built-in equation AVG3

	1	2	3	4	5	6	7	8	9	10
TRESP3	1	SUMM*3	AVG3	4						
+	TVAR	1	2	3	4					
+	TRESP1	10	20	30	40	50	60	70	80	
+		90	100							

Field Information Description

2	ID	Unique identification number. (Integer > 0). See remark 6.
3	LABEL	User defined label for information only.
4	LIBID or NAME	Library identification number. (Integer > 0). or one of the following built-in function names: SUM1, SUM2, SUM3, SUM4, AVG1, AVG2, AVG3, AVG4, NORM1, NORM2, NORM3, NORM4, MULT1, MULT2, MULT3, MULT4, SAVG1, SAVG2, SAVG3, SAVG4, ASAVG1, ASAVG2, ASAVG3, ASAVG4, PNORM2 or SDEV. See remark 18.
5	REGION	Region identifier for constraint screening. (Integer > 0 or blank. Default puts this set of responses in a unique region.).
2	TVAR	Word indicating TVAR identification numbers.
3,4,..	NDVi	TVAR identification number. (Integer > 0 or Blank or "THRU").
2	DTABLE	Word indicating constant identification numbers in DTABLE.
3,4,..	NCj	Name of constant defined in DTABLE input. (Character). See Remark 19.
2	TRESP1	Word indicating TRESP1 identification numbers.
3,4,..	NRk	TRESP1 identification number. (Integer > 0 or Blank or "THRU"). See remarks 2, 4 and 5.
2	TRESP1L	Word indicating TRESP1 and Loadcase identification numbers.
3,5,..	NRk	TRESP1 identification number (Integer > 0 or Blank). See remark 12.
4,6,..	LIDRk	Loadcase identification number (Integer > 0).

2	TRESP2	Word indicating TRESP2 or TRESP3 identification numbers.
3,4,..	NSk	TRESP2 or TRESP3 identification number. (Integer > 0 or Blank or "THRU").

Remarks:

1. TRESP3 entries can reference TVAR, DTABLE, TRESP1, TRESP2 and TRESP3 entries.
2. If the TRESP1 keyword is used, then the TRESP1 responses referenced by the TRESP3 entry must all be of the same analysis type (static, frequency, direct dynamic, modal dynamic, random, heat transfer or buckling). Static, frequency, heat transfer and buckling responses cannot be mixed. Mass and volume responses cannot be mixed with static, frequency, direct dynamic, modal dynamic, heat transfer or buckling responses.
3. Different static responses **can** be mixed on a single TRESP3 entry. For example, strain energy and displacement responses can both be referenced by a single TRESP3 entry.
4. TRESP3 can refer to TRESP1s that have multiple physical responses associated with them. However, each referenced TRESP1 has to have the same number of responses. For PTYPE = PROP, the number of responses is the sum of the elements associated with each property entered in the referenced TRESP1.
5. For RTYPE = DISP, the number of ATTi may be different on each TRESP1 entry, but the number of ATTi multiplied by the number of components must be the same. DISP responses on the TRESP1 entries can reference different grids and components (directions).
6. TRESP3 entries must have unique identification numbers with respect to **TRESP1** and **TRESP2** entries.
7. The sequence in which TVAR, DTABLE and TRESP1 / TRESP1L / TRESP2 occur is not arbitrary. They must be in the order shown. Any of these five words along with the identification numbers associated with them can be omitted if they are not involved in this TRESP3 relationship. However, at least one of the words TVAR, TRESP1 / TRESP1L / TRESP2 should exist.
8. TRESP3 entries that reference TRESP1 entries which are mass or volume responses will be in a distinct REGION. Two different TRESP3 entries that reference static, frequency, direct dynamic, modal dynamic heat transfer and/or buckling responses will never be in the same REGION, even if they are assigned the same REGION identification number. TRESP1 and TRESP3 responses will never be contained in the same region, even if they are assigned the same REGION identification number. The default is to put all responses referenced by one TRESP3 entry in a distinct region.
9. If a TRESP3 entry is not referenced by a TOBJ, TCONS or TINDEX entry, this response is calculated for all load cases and printed with other TRESP3 values in the TRESP2/3 response table.

10. The variables identified by NDVi, NCj, NRk and NSk correspond to values in the VAR array in the GNSUBi user routine. The parameters are assumed to be; first, according to the order of TVAR, DTABLE and TRESP1 / TRESP2 specifications, and second, NDVi, NCj and NRk / NSk. In the example above, the order is: TVAR 87, 25, etc.; then DTABLE 1, 8, etc.; and finally TRESP1 12, 5, etc.
11. If the TRESP1L keyword is used, the responses from different analysis types; static, frequency, direct dynamic, modal dynamic, heat transfer, and/or buckling, can be mixed.
12. If the TRESP1L keyword is used, then the LID on the TCONS data must be blank.
13. If the TRESP1L keyword is used, then the LID on the TOBJ data must be the same as the LID of the first TRESP1 referenced (LIDR1).
14. If the TRESP1L keyword is used and the response type is MASSFR or INERTIA, then the LID should be the ID of the first LOADCASE.
15. If a library ID is given, the **DRESP3** executive control command must be used to identify the shared object (DLL) that contains the user calculation subroutines. See **Use of the DRESP3 / TRESP3 capability** (p. 307) for a description of how to create the DRESP3 interface function.
16. If the TRESP1L keyword is used with dynamic responses, then each dynamic loadcase referenced (LID_i) must have the same number of loading frequencies.
17. The built-in functions are defined as follows (X1, ..., XN are all of the TVAR, DTABLE, and TRESP1 / TRESP1L / TRESP2 argument values; N is the number of arguments; LBi and UBi are the corresponding lower bound and upper bound from the TVAR entry for TVAR arguments or LBi = 0.0 and UBi = 1.0 for all other arguments types):

$$\begin{aligned} \text{AVG1} &= (X1 + X2 + \dots + XN)/N \\ \text{AVG2} &= (X1^{**2} + X2^{**2} + \dots + XN^{**2})/N \\ \text{AVG3} &= (X1^{**3} + X2^{**3} + \dots + XN^{**3})/N \\ \text{AVG4} &= (X1^{**4} + X2^{**4} + \dots + XN^{**4})/N \end{aligned}$$

$$\begin{aligned} \text{MULT1} &= X1 * X2 * \dots * XN \\ \text{MULT2} &= \text{MULT1}^{**2} \\ \text{MULT3} &= \text{MULT1}^{**3} \\ \text{MULT4} &= \text{MULT1}^{**4} \end{aligned}$$

$$\begin{aligned} \text{NORM1} &= (\text{ABS}(X1) + \text{ABS}(X2) + \dots + \text{ABS}(XN)) \\ \text{NORM2} &= (\text{ABS}(X1)^{**2} + \text{ABS}(X2)^{**2} + \dots + \text{ABS}(XN)^{**2})^{**}(1/2) \\ \text{NORM3} &= (\text{ABS}(X1)^{**3} + \text{ABS}(X2)^{**3} + \dots + \text{ABS}(XN)^{**3})^{**}(1/3) \\ \text{NORM4} &= (\text{ABS}(X1)^{**4} + \text{ABS}(X2)^{**4} + \dots + \text{ABS}(XN)^{**4})^{**}(1/4) \end{aligned}$$

$$\begin{aligned} \text{SUM1} &= X1 + X2 + \dots + XN \\ \text{SUM2} &= X1^{**2} + X2^{**2} + \dots + XN^{**2} \end{aligned}$$

$$\text{SUM3} = X1^{**3} + X2^{**3} + \dots + XN^{**3}$$

$$\text{SUM4} = X1^{**4} + X2^{**4} + \dots + XN^{**4}$$

$$\text{SAVG1} = ((X1-LB1)/(UB1-LB1) + (X2-LB2)/(UB2-LB2) + \dots + (XN-LBN)/(UBN-LBN))/N$$

$$\text{SAVG2} = (((X1-LB1)/(UB1-LB1))^{**2} + ((X2-LB2)/(UB2-LB2))^{**2} + \dots + (XN-LBN)/(UBN-LBN))^{**2}/N$$

$$\text{SAVG3} = (((X1-LB1)/(UB1-LB1))^{**3} + ((X2-LB2)/(UB2-LB2))^{**3} + \dots + (XN-LBN)/(UBN-LBN))^{**3}/N$$

$$\text{SAVG4} = (((X1-LB1)/(UB1-LB1))^{**4} + ((X2-LB2)/(UB2-LB2))^{**4} + \dots + (XN-LBN)/(UBN-LBN))^{**4}/N$$

$$\text{ASAVG1} = (\text{ABS}(X1)/\text{MAX}(\text{ABS}(UB1),\text{ABS}(LB1)) + \text{ABS}(X2)/\text{MAX}(\text{ABS}(UB2),\text{ABS}(LB2)) + \dots + \text{ABS}(XN)/\text{MAX}(\text{ABS}(UBN),\text{ABS}(LBN)))/N$$

$$\text{ASAVG2} = ((\text{ABS}(X1)/\text{MAX}(\text{ABS}(UB1),\text{ABS}(LB1)))^{**2} + (\text{ABS}(X2)/\text{MAX}(\text{ABS}(UB2),\text{ABS}(LB2)))^{**2} + \dots + (\text{ABS}(XN)/\text{MAX}(\text{ABS}(UBN),\text{ABS}(LBN)))^{**2})/N$$

$$\text{ASAVG3} = ((\text{ABS}(X1)/\text{MAX}(\text{ABS}(UB1),\text{ABS}(LB1)))^{**3} + (\text{ABS}(X2)/\text{MAX}(\text{ABS}(UB2),\text{ABS}(LB2)))^{**3} + \dots + (\text{ABS}(XN)/\text{MAX}(\text{ABS}(UBN),\text{ABS}(LBN)))^{**3})/N$$

$$\text{ASAVG4} = ((\text{ABS}(X1)/\text{MAX}(\text{ABS}(UB1),\text{ABS}(LB1)))^{**4} + (\text{ABS}(X2)/\text{MAX}(\text{ABS}(UB2),\text{ABS}(LB2)))^{**4} + \dots + (\text{ABS}(XN)/\text{MAX}(\text{ABS}(UBN),\text{ABS}(LBN)))^{**4})/N$$

$$\text{PNORM2} = \text{SQRT} (\text{MAX}(X1,0.0)^{**2} + \text{MAX}(X2,0.0)^{**2} + \dots + \text{MAX}(XN,0.0)^{**2})$$

$$\text{SDEV} = \text{SQRT} (((X1-AVG1)^{**2} + (X2-AVG1)^{**2} + \dots + (XN-AVG1)^{**2})/N)$$

18. The name CYCLE can be used in the DTABLE list to pass the current value of the design cycle number (as long as the name CYCLE is not explicitly defined on the DTABLE entry).
19. NDVi, NRk and NSk can be an integer or a string "THRU". If a NDVi is "THRU", it specifies a sequential list of TVAR; if a NRk is "THRU", it specifies a sequential list of TRESP1; if a NSk is "THRU", it specifies a sequential list of TRESP2 or TRESP3.

9.2.16 TSELECT

Data Entry: **TSELECT** - Select a fraction of topology design variables to keep.

Description: Create constraints such that a given fraction of the total number of listed topology extra variables ('TVAR'), or internally generated topology design variables, move to their upper bound while the rest of the variables move to their lower bound. The internally generated design variables are associated to a topology region corresponding to properties listed ('PROP').

Format:

1	2	3	4	5	6	7	8	9	10
TSELECT	ID	LABEL	FRACT	BTYPE	GTYPE	TOL			
+	"TVAR"	NDV1	NDV2	NDV3	NDV4	NDV5	NDV6	NDV7	NDV8
+		NDV9	-etc.-						

Alternate Format:

1	2	3	4	5	6	7	8	9	10
TSELECT	ID	LABEL	FRACT	BTYPE	GTYPE	TOL			
+	"PROP"	PID1	PID2	PID3	PID4	PID5	PID6	PID7	PID8
+		PID9	-etc.-						

Example 1: Create constraints so that 50% of the listed topology extra variables move to their upper bound and 50% move to the lower bound.:

1	2	3	4	5	6	7	8	9	10
TSELECT	1	PartFrac	0.5	EXACT	AVG				
+	TVAR	87	25	3	8				

Example 2: Create constraints so that 30% of the internally created topology design variables, that are associated to property 100, move to their upper bound. The rest of the design variable will be pushed to their lower bound.:

1	2	3	4	5	6	7	8	9	10
TSELECT	1	PartFrac	0.3	EXACT					
+	PROP	100							

Remarks:

Field	Information	Description
2	ID	Unique TSELECT identification number. (Integer > 0).
3	LABEL	User defined label for information only.
4	FRACT	Maximum or Exact average of design variable values. (0.0<Real <=1.0)
5	BTYPE	One of the following types: UPPER or EXACT or blank. Default = EXACT. See remark 1.
6	GTYPE	One of the following types: AVG or ABS or blank. Default = AVG. See remark 2.
7	TOL	Tolerance for satisfying selection constraints. Default = 0.005.
2	TVAR	Word indicating TVAR identification numbers.
3,4,..	NDVi	TVAR identification number. (Integer > 0 or "THRU").
2	PROP	Word indicating property identification numbers.
3,4,..	PIDi	Property identification number. (Integer > 0 or "THRU"). The properties listed must also be listed in a TPROP entry

1. A TSELECT entry with the BTYPE=EXACT option creates two internal constraints using the listed design variable entries so that a given fraction of them move to their upper bound (GTYPE=AVG) or either bound (GTYPE = ABS) while moving the rest to their lower bound or 0.0. The constraint types are controlled by GTYPE. FRACT controls the fraction of the number of the design variables that are desired to move to their upper bound. A TSELECT entry with the BTYPE=UPPER option creates one internal constraint using the listed design variable entries so that a special average of them does not exceed a given fraction value.
2. If GTYPE = AVG or blank, the following function is used:

$$G = ((X1-LB1)/(UB1-LB1) + (X2-LB2)/(UB2-LB2) + \dots + (XN-LBN)/(UBN-LBN))/N$$

If GTYPE = ABS, the following function is used:

$$G = (ABS(X1)/MAX(ABS(UB1),ABS(LB1)) + ABS(X2)/MAX(ABS(UB2),ABS(LB2)) + \dots + ABS(XN)/MAX(ABS(UBN),ABS(LBN)))/N$$
3. If BTYPE = UPPER, the following constraint is used:

$$G \leq FRACT + TOL$$

If BTYPE = EXACT, two constraints will be internally constructed in *GENESIS* so that

$$FRACT - TOL \leq G \leq FRACT + TOL$$
4. A TSELECT entry may have either a TVAR continuation or a PROP continuation, but not both.
5. Multiple TSELECT are allowed.

6. Each property listed in a PROP continuation must be topology-designed with a corresponding TPROP entry.
7. NDVi and PIDi can be an integer or a string “THRU”. If a NDVi is “THRU”, it specifies a sequential list of TVAR; if a PIDi is “THRU”, it specifies a sequential list of property IDs.

9.2.17 TSYM1

Data Entry: **TSYM1** - Fabrication Constraints for Topology Optimization, Form 1.

Description: Define fabrication constraints on a coordinate system defined by three grids.

Format:

1	2	3	4	5	6	7	8	9	10
TSYM1	ID	G1	G2	G3	n	NSECT	DISTX	DISTY	DISTZ
+	TYPE1	TYPE2	TYPE3						
+	SYMV1	SYMV2	SYMV3	SYMV4					
+	STHICK	PUNCH	VOIDDNS	OFFSET					

Example 1: Filling in +x direction, minimum member size = 5.0 mm:

1	2	3	4	5	6	7	8	9	10
TSYM1	101	7	18	15					
+	FBX								
+	5.0								

Example 2: Triple mirror symmetry conditions:

1	2	3	4	5	6	7	8	9	10
TSYM1	1025	17	18	19					
+	MYZ	MZX	MYX						

Example 3: Extrusion along x direction and mirror symmetries on ZX and XY planes:

1	2	3	4	5	6	7	8	9	10
TSYM1	103	447	44	501					
+	EX	MZX	MYX						

Example 4: Minimum member size = 5.0 mm:

1	2	3	4	5	6	7	8	9	10
TSYM1	104	7	18	15					
+									
+	5.0								

Example 5: Smoothed minimum member size = $5.0 = (\text{SYMV1} = 3.0) + 2 * (\text{SYMV1} = 1.0)$ mm:

1	2	3	4	5	6	7	8	9	10
TSYM1	104	7	18	15					
+									
+	3.0	1.0							

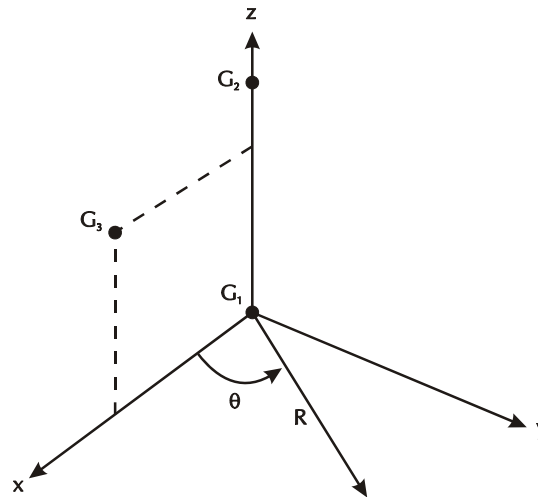
Field	Information	Description
2	ID	Unique TSYM identification number. (Integer > 0).
3, 4, 5,	G1, G2, G3	Grid point identification number. (Integer > 0; $G1 \neq G2 \neq G3$).
6	n	Number of cyclic symmetry sections (Integer > 1 or Blank).
7	NSECT	Number of sectors. Used with RBX, RBY, RBZ, RTX, RTY, RTZ, RGX, RGY, RGZ, KX, KY or KZ (Integer > 1 or Blank). Default = 36 or $6*n$ when n is not zero.
8,9,10	DISTX,DISTY, DISTZ	Pattern pitch distance X (or Y or Z). (Real > 0 or blank). A positive value will cause the software to generate a design proposal which features are periodic in the X (or Y or Z) direction. The pattern pitch is the distance where the patterns repeats. Pattern pitches are used together with PX, PY, PZ, P0X, P0Y and/or P0Z. See Remarks 9, 10 and 11.
2	TYPE1	Fabrication constraint type. One of "MYZ", "MZX", "MXY", "CX", "CY", "CZ", "EX", "EY", "EZ", "FBX", "FBY", "FBZ", "FTX", "FTY", "FTZ", "FSX", "FSY", "FSZ", "FGX", "FGY", "FGZ", "F0X", "F0Y", "F0Z", "PX", "PY", "PZ", "P0X", "P0Y", "P0Z", "RBX", "RBY", "RBZ", "RTX", "RTY", "RTZ", "RGX", "RGY", "RGZ", "KX", "KY" or "KZ", "SBX", "SBY", "SBZ", "STX", "STY", "STZ", "S2X", "S2Y", "S2Z", "UYZ", "UZX", "UXY", or "UXYZ", .
3,4	TYPE2, TYPE3	Fabrication constraint type. One of "MYZ", "MZX", "MXY", "CX", "CY", "CZ", "EX", "EY", "EZ", "FBX", "FBY", "FBZ", "FTX", "FTY", "FTZ", "FSX", "FSY", "FSZ", "FGX", "FGY", "FGZ", "F0X", "F0Y", "F0Z", "PX", "PY", "PZ", "P0X", "P0Y", "P0Z", "RBX", "RBY", "RBZ", "RTX", "RTY", "RTZ", "RGX", "RGY", "RGZ", "KX", "KY", "KZ", "SBX", "SBY", "SBZ", "STX", "STY", "STZ", "S2X", "S2Y", "S2Z", "UYZ", "UZX", "UXY", or "UXYZ", or Blank. See Remarks 6 and 7.
2	SYMV1	Pole spacing. Required for FBX, FBY, FBZ, FTX, FTY, FTZ, FSX, FSY, FSZ, FGX, FGY, FGZ, F0X, F0Y, F0Z, RBX, RBY, RBZ, RTX, RTY, RTZ, RGX, RGY, RGZ, KX, KY, KZ, SBX, SBY, SBZ, STX, STY, STZ, S2X, S2Y or S2Z, (Real>0.0). Optional for EX, EY, EZ, PX, PY, PZ, P0X, P0Y, P0Z, .
3	SYMV2	Spread size. Optionally used with FBX, FBY, FBZ, FTX, FTY, FTZ, FSX, FSY, FSZ, FGX, FGY, FGZ, F0X, F0Y, F0Z, EX, EY, EZ, PX, PY, PZ, P0X, P0Y, P0Z, RBX, RBY, RBZ, RTX, RTY, RTZ, RGX, RGY, RGZ, KX, KY or KZ, SBX, SBY, SBZ, STX, STY, STZ, S2X, S2Y or S2Z, (Real≥0.0 or Blank). Default=0.0.
3	SYMV3	Maximum member size. Optionally used with FBX, FBY, FBZ, FTX, FTY, FTZ, FSX, FSY, FSZ, FGX, FGY, FGZ, F0X, F0Y, F0Z, EX, EY or EZ, PX, PY, PZ, P0X, P0Y, P0Z, RBX, RBY, RBZ, RTX, RTY, RTZ, RGX, RGY, RGZ, KX, KY or KZ, SBX, SBY, SBZ, STX, STY, STZ, S2X, S2Y or S2Z, (Real>0.0). SYMV3 requires SYMV1 to be non blank.

4	SYMV4	Minimum Gap. Optionally used with FBX, FBY, FBZ, FTX, FTY, FTZ, FSX, FSY, FSZ, FGX, FGY, FGZ, F0X, F0Y, F0Z, EX, EY, EZ, PX, PY, PZ, P0X, P0Y, P0Z, RBX, RBY, RBZ, RTX, RTY, RTZ, RGX, RGY, RGZ, KX, KY, KZ, SBX, SBY, SBZ, STX, STY, STZ, S2X, S2Y or S2Z, (Real>0.0 or Blank). Default=SYMV3. SYMV4 requires SYMV3 to be non blank.
2	STHICK	Thickness of the stampable sheet(s). Required for SBX, SBY, SBZ, STX, STY, STZ, S2X, S2Y, S2Z. Optional for FBX, FBY, FBZ, FTX, FTY, FTZ, FGX, FGY, FGZ. (Real > 0 or -1.0≤Real<0). A positive value enters the exact value of the thickness. A negative value will cause the program to calculate the thickness as a fraction of the local height. The fraction is the absolute value of STHICK.
3	PUNCH	Sheet/Casting integrity. Optionally used with SBX, SBY, SBZ, STX, STY, STZ, S2X, S2Y, S2Z, FBX, FBY, FBZ, FTX, FTY, FTZ, FGX, FGY, FGZ. ("YES" or "NO" or Blank). YES will allow holes though the sheet. NO will not allow holes through the sheet. Default is YES.
4	VOIDDNS	Density of the void area. Optionally used with SBX, SBY, SBZ, STX, STY, STZ, S2X, S2Y, S2Z (Real>0.0 or Blank) Default=0.001.
5	OFFSET	Starting location of the sheet. Optionally used with SBX, SBY, SBZ, STX, STY, STZ, S2X, S2Y, S2Z (0≤Real≤1.0 or Blank). OFFSET is a fraction of the local height away from the top and/or bottom. Default=0.0.

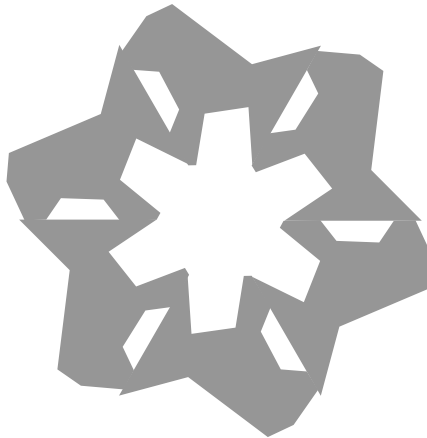
Remarks:

1. TSYM1 entries must be referenced by **TPROP** to be used.
2. The first continuation line must be provided. The second continuation line must be provided if any TYPEi is FBX, FBY, FBZ, FTX, FTY, FTZ, FGX, FGY, FGZ, F0X, F0Y, F0Z, FSX, FSY, FSZ, RBX, RBY, RBZ, RTX, RTY, RTZ, RGX, RGY, RGZ, KX, KY or KZ. The second and third continuation lines must be provided if any TYPEi is SBX, SBY, SBZ, STX, STY, STZ, S2X, S2Y or S2Z.

3. The three grids define three directions X, Y and Z. These three grids cannot be colinear. These grids do not have to be connected to the structure.



4. If CX, CY or CZ are used, then field n must be given, and n cyclically symmetric sections are created about specified axis. For example if $n = 6$, a solution might look like:



5. Up to three fabrication constraints are allowed per entry. Following is a table with available fabrication constraints. See [Filling Types](#) (p. 390) for an illustration of the filling constraint types.

TYPE	Description of Fabrication Constraints
MXY	Mirror symmetry with respect to the XY plane
MYZ	Mirror symmetry with respect to the YZ plane
MZX	Mirror symmetry with respect to the ZX plane

CX	Cyclic symmetry about the X axis
CY	Cyclic symmetry about the Y axis
CZ	Cyclic symmetry about the Z axis
EX	Extrusion along the X axis
EY	Extrusion along the Y axis
EZ	Extrusion along the Z axis
UXY	Uniform in planes parallel to the XY plane
UYZ	Uniform in planes parallel to the YZ pane
UZX	Uniform in planes parallel to the ZX plane
UXYZ	Uniform in all directions
PX	Periodic pattern repetition in the X direction (with user specified pitch pattern distance DISTX)
PY	Periodic pattern repetition in the Y direction (with user specified pitch pattern distance DISTY)
PZ	Periodic pattern repetition in the Z direction (with user specified pitch pattern distance DISTZ)
P0X	Symmetric Periodic pattern repetition in the X and -X directions (with user specified pitch pattern distance DISTX)
P0Y	Symmetric Periodic pattern repetition in the Y and -Y directions (with user specified pitch pattern distance DISTY)
P0Z	Symmetric Periodic pattern repetition in the Z and -Z directions (with user specified pitch pattern distance DISTZ)
FBX	Filling in the +X direction
FBY	Filling in the +Y direction
FBZ	Filling in the +Z direction
FTX	Filling in the -X direction
FTY	Filling in the -Y direction
FTZ	Filling in the -Z direction
FSX	Filling simulatneously from +X and -X directions (filling from the outside in)

FSY	Filling simulatneously from +Y and -Y directions (filling from the outside in)
FSZ	Filling simulatneously from +Z and -Z directions (filling from the outside in)
FGX	Filling on +X and -X directions from general surface (filling from the inside out)
FGY	Filling on +Y and -Y directions from general surface (filling from the inside out)
FGZ	Filling on +Z and -Z directions from general surface (filling from the inside out)
F0X	Filling symmetrically from X=0.0 toward the top and bottom (filling from the inside out)
F0Y	Filling symmetrically from Y=0.0 toward the top and bottom (filling from the inside out)
F0Z	Filling symmetrically from Z=0.0 toward the top and bottom (filling from the inside out)
RBX	Radial Filling From the Inner Surface about the X axis
RBZ	Radial Filling From the Inner Surface about the Z axis
RTX	Radial Filling From the Outer Surface about the X axis
RTY	Radial Filling From the Outer Surface about the Y axis
RTZ	Radial Filling From the Outer Surface about the Z axis
RGX	Radial Filling From the General Surface about the X axis (from the interior to the inner and outer surfaces)
RGY	Radial Filling From the General Surface about the Y axis (from the interior to the inner and outer surfaces)
RGZ	Radial Filling From the General Surface about the Z axis (from the interior to the inner and outer surfaces)
KX	Radial Spokes about the X axis
KY	Radial Spokes about the Y axis
KZ	Radial Spokes about the Z axis
SBX	Sheet forming normal to +X (1 Layer) (initial configuration starts at the bottom)

SBY	Sheet forming normal to +Y (1 Layer) (initial configuration starts at the bottom)
SBZ	Sheet forming normal to +Z (1 Layer) (initial configuration starts at the bottom)
STX	Sheet forming normal to +X (1 Layer) (initial configuration starts at the top)
STY	Sheet forming normal to +Y (1 Layer) (initial configuration starts at the top)
STZ	Sheet forming normal to +Z (1 Layer) (initial configuration starts at the top)
S2X	Sheet forming normal to +X (2 Layer) (initial configuration of first layer starts at the top; initial configuration of second layer starts at the bottom)
S2Y	Sheet forming normal to +Y (2 Layer) (initial configuration of first layer starts at the top; initial configuration of second layer starts at the bottom)
S2Z	Sheet forming normal to +Z (2 Layer) (initial configuration of first layer starts at the top; initial configuration of second layer starts at the bottom)

- Two types of symmetries method are used internally in the program: Element based symmetry or location based symmetries. Element based symmetries enforce symmetries using symmetry of the elements in the mesh. Location based symmetry enforce symmetry based on symmetry of the space. Location based symmetries requires the SYMV1 field be specified, and are currently available for extrusions, filling and stamping. For location based symmetries , the minimum member size is approximately $SYMV1 + 2*SYMV2$.
- Not all fabrication constraints can be mixed together. Following is a table with the allowed combinations (excluding periodic):

$MXY+MYZ, MXY+MZX, MYZ+MZX, MXY+MYZ+MZX$
$CX+MYZ, CY+MZX, CZ+MXY$
$EX+MXY, EX+MZX, EX+MXY+MZX$ $EY+MYZ, EY+MXY, EY+MYZ+MXY$ $EZ+MZX, EZ+MYZ, EZ+MZX+MYZ$
$EX+CX, EY+CY, EZ+CZ$
$UYZ+MYZ, UZX+MZX, UXY+MXY$

FiX+MXY, FiX+MZX, FiX+MXY+MZX FiY+MYZ, FiY+MXY, FiY+MYZ+MXY FiZ+MZX, FiZ+MYZ, FiZ+MZX+MYZ
FiX+CX, FiY+CY, FiZ+CZ
RiX+MXY, RiX+MYZ, RiX+ MZX, RiX+MXY+MYZ,RiX+MXY+MZX,RiX+MYZ+MZX RiY+MXY, RiY+MYZ, RiY+ MZX, RiY+MXY+MYZ,RiY+MXY+MZX,RiY+MYZ+MZX RiZ+MXY, RiZ+MYZ, RiZ+ MZX, RiZ+MXY+MYZ,RiZ+MXY+MZX,RiZ+MYZ+MZX
RiX+CX, RiY+CY, RiZ+CZ
RiX+EX, RiY+EY, RiZ+EZ
RiX+CX+EX, RiY+CY+EY, RiZ+CZ+EZ
RiX+CX+MYZ, RiY+CY+MZX, RiZ+CZ+MXY
RiX+EX+MXY, RiX+EX+MZX, RiY+EY+MXY,RiY+EY+MYZ, RiZ+EZ+MYZ, RiZ+EZ+MZX,

8. In the above table:
 FiX is one of FBX, FTX, FSX, FGX, F0X, SBX, STX or S2X
 FiY is one of FBY, FTY, FSY, FGY, F0Y, SBY, STY or S2Y
 FiZ is one of FBZ, FTZ , FSZ, FGZ, F0Z, SBZ, STZ or S2Z
 RiX is one of RBX, RTX, RGX, or KX
 RiY is one of RBY, RTY, RGY,or KY
 RiZ is one of RBZ, RTZ, RGZ, or KZ
9. Periodic structures with repetetions on the x directions requires using PX (or P0X) and DISTX. Periodic structures with repetetions on the y directions requires using PY (or P0Y) and DISTY. Periodic structures with repetetions on the z directions requires using PZ (or P0Z) and DISTZ.
10. Periodic fabrication constraints and their valids combination with other fabrication constraints are

Types that can combined with Periodic	Combinations	Number of combinati ons
2 or 3 Periodic	PX+PY, PX+PZ, PY+PZ, PX+PY+PZ	4
Periodic + 1 or 2 Mirrors	PX+MXY, PX+MZX,PX+MXY+MZX, PY+MXY, PY+MYZ, PY+MXY+MYZ, PZ+MZX, PZ+MYZ, PZ+MZX+MYZ	9
2 Periodic + Mirror	PX+PY +MXY, PX+PZ +MZX, PY+PZ +MYZ	3

Periodic + Cyclic	PX+CX, PY+CY, PZ+CZ	3
Periodic + Extrusion	PX+EY, PX+EZ, PY+EX, PY+EZ, PZ+EX, PZ+EY	6
2 Periodic + Extrusion	PX+PY+EZ, PX+PZ+EY, PY+PZ+EX	3
Periodic + Mirror + Extrusion	PX+MX+Y+EY, PX+MZ+X+EZ, PY+MX+Y+EX, PY+MY+Z+EZ, PZ+MZ+X+EX, PZ+MY+Z+EY	6
Periodic + Uniform	PX+UYZ, PY+UZX, PZ+UXY	3
Periodic + Filling	PX+FiY, PX+FiZ, PY+FiX, PY+FiZ, PZ+FiX, PZ+FiY	6
2 Periodic + Filling	PX+PY+FiZ, PX+PZ+FiY, PY+PZ+FiX,	3
Periodic + Mirror + Filling	PX+MX+Y+FiY, PX+MZ+X+FiZ, PY+MX+Y+FiX, PY+MY+Z+FiZ, PZ+MZ+X+FiX, PZ+MY+Z+FiY	6
Periodic + Radial	RiX+PX, RiY+PY, RiZ+ PZ	3
Periodic + Mirror + Radial	PX+MX+Y+RiX, PX+MZ+X+RiX, PY+MX+Y+RiY, PY+MY+Z+RiY, PZ+MY+Z+RiZ, PZ+MZ+X+RiZ	6
Periodic + Cyclic + Radial	PX+CX+RiX PY+CY+RiY, PZ+CZ+RiZ	3

11. In the above table:

FiX is one of FBX, FTX, FSX, FGX, F0X, SBX, STX or S2X

FiY is one of FBY, FTY, FSY, FGY, F0Y, SBY, STY or S2Y

FiZ is one of FBZ, FTZ, FSZ, FGZ, F0Z, SBZ, STZ or S2Z

RiX is one of RBX, RTX, RGX, or KX

RiY is one of RBY, RTY, RGY, or KY

RiZ is one of RBZ, RTZ, RGZ, or KZ

PX is one of PX or P0X

PY is one of PY or P0Y

PZ is one of PZ or P0Z

12. Minimum member size can be obtained without the need of symmetries, the minimum member size is approximately $SYM V1 + 2 * SYM V2$.

13. UXYZ can not be mixed with any other fabrication constraints.

9.2.18 TSYM2

Data Entry: **TSYM2** -Fabrication Constraints for Topology Optimization, Form 2.

Description: Define fabrication constraints on a coordinate system defined by three points.

Format:

1	2	3	4	5	6	7	8	9	10
TSYM2	ID	RID	A1	A2	A3	B1	B2	B3	
+	C1	C2	C3	n	NSECT	DISTX	DISTY	DISTZ	
+	TYPE1	TYPE2	TYPE3						
+	SYMV1	SYMV2	SYMV3	SYMV4					
+	STHICK	PUNCH	VOIDDNS	OFFSET					

Example 1: Filling on +x direction, minimum member size = 5.0 mm:

1	2	3	4	5	6	7	8	9	10
TSYM2	201		0.0	0.0	0.0	0.0	0.0	1.0	
+	1.0	0.0	0.0						
+	FX								
+	5.0								

Example 2: Triple mirror symmetry conditions:

1	2	3	4	5	6	7	8	9	10
TSYM2	202		2.0	3.0	0.0	0.0	1.0	1.0	
+	1.0	-1.0	2.0						
+	MYZ	MZX	MXY						

Example 3: Extrusion on x direction and mirror symmetries on ZX and XY planes:

1	2	3	4	5	6	7	8	9	10
TSYM2	203		2.0	3.0	0.0	0.0	1.0	1.0	
+	1.0	-1.0	2.0						
+	EX	MZX	MXY						

Example 4: Smoothed minimum member size = $5.0 = (\text{SYMV1}=3.0) + 2 * (\text{SYMV1}=1.0)$ mm:

1	2	3	4	5	6	7	8	9	10
TSYM2	204	7	18	15					
+									
+	3.0	1.0							

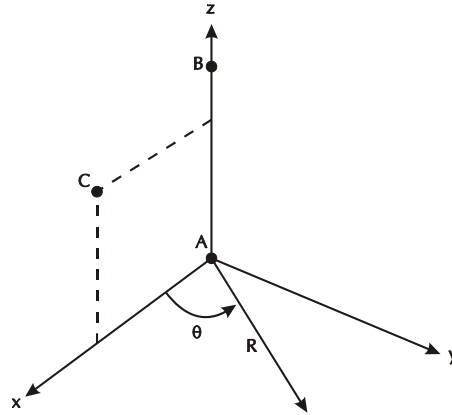
Field	Information	Description
2	ID	Unique TSYM identification number. (Integer > 0).
3	RID	Reference to a coordinate system. (Integer ≥ 0 or Blank).
4, 5, 6, 7, 8, 9 2, 3, 4	A1, A2, A3 B1, B2, B3 C1, C2, C3	Coordinates of three points in the coordinate system defined in field 3. (Real or Blank. Default = 0.0).
5	n	Number of cyclic symmetry sections. (Integer > 1 or Blank).
6	NSECT	Number of sectors. Used with RBX, RBY, RBZ, RTX, RTY, RTZ, RGX, RGY, RGZ, KX, KY or KZ (Integer > 1 or Blank). Default = 36 or 6*n when n is not zero.
7,8,9	DISTX,DISTY, DISTZ	Pattern pitch distance X (or Y or Z). (Real > 0 or blank). A positive value will cause the software to generate a design proposal which features are periodic in the X (or Y or Z) direction. The pattern pitch is the distance where the patterns repeats. Pattern pitches are used together with PX, PY, PZ, P0X, P0Y and/or P0Z. See Remarks 10, 11 and 12.
2	TYPE1	Fabrication constraint type. One of "MYZ", "MZX", "MXY", "CX", "CY", "CZ", "EX", "EY", "EZ", "FBX", "FBY", "FBZ", "FTX", "FTY", "FTZ", "FSX", "FSY", "FSZ", "FGX", "FGY", "FGZ", "F0X", "F0Y", "F0Z", "PX", "PY", "PZ", "P0X", "P0Y", "P0Z", "RBX", "RBY", "RBZ", "RTX", "RTY", "RTZ", "RGX", "RGY", "RGZ", "KX", "KY", "KZ", "SBX", "SBY", "SBZ", "STX", "STY", "STZ", "S2X", "S2Y", "S2Z", "UYZ", "UZX", "UXY" or "UXYZ" ,
3,4	TYPE2, TYPE3	Fabrication constraint type. One of "MYZ", "MZX", "MXY", "CX", "CY", "CZ", "EX", "EY", "EZ", "FBX", "FBY", "FBZ", "FTX", "FTY", "FTZ", "FSX", "FSY", "FSZ", "FGX", "FGY", "FGZ", "F0X", "F0Y", "F0Z", "PX", "PY", "PZ", "P0X", "P0Y", "P0Z", "RGX", "RGY", "RGZ", "KX", "KY", "KZ", "SBX", "SBY", "SBZ", "STX", "STY", "STZ", "S2X", "S2Y", "S2Z", "UYZ", "UZX", "UXY" or "UXYZ" or Blank. See Remarks 6 and 7.
2	SYMV1	Pole spacing. Required for FBX, FBY, FBZ, FTX, FTY, FTZ, FSX, FSY, FSZ, FGX, FGY, FGZ, F0X, F0Y, F0Z, RBX, RBY, RBZ, RTX, RTY, RTZ, RGX, RGY, RGZ, KX, KY, KZ, SBX, SBY, SBZ, STX, STY, STZ, S2X, S2Y or S2Z, (Real>0.0). Optional for EX, EY, EZ, PX, PY, PZ, P0X, P0Y, or P0Z, .
3	SYMV2	Spread size. Optionally used with FBX, FBY, FBZ, FTX, FTY, FTZ, FSX, FSY, FSZ, FGX, FGY, FGZ, F0X, F0Y, F0Z, EX, EY, EZ, PX, PY, PZ, P0X, P0Z, P0Z, RBX, RBY, RBZ, RTX, RTY, RTZ, RGX, RGY, RGZ, KX, KY or KZ, SBX, SBY, SBZ, STX, STY, STZ, S2X, S2Y or S2Z, (Real \geq 0.0 or Blank). Default=0.0.

3	SYMV3	Maximum member size. Optionally used with FBX, FBY, FBZ, FTX, FTY, FTZ, FSX, FSY, FSZ, FGX, FGY, FGZ, F0X, F0Y, F0Z, EX, EY, EZ, PX, PY, PZ, P0X, P0Y, P0Z, RBX, RBY, RBZ, RTX, RTY, RTZ, RGX, RGY, RGZ, KX, KY or KZ, SBX, SBY, SBZ, STX, STY, STZ, S2X, S2Y or S2Z, (Real>0.0). SYMV3 requires SYMV1 to be non blank.
4	SYMV4	Minimum Gap. Optionally used with FBX, FBY, FBZ, FTX, FTY, FTZ, FSX, FSY, FSZ, FGX, FGY, FGZ, F0X, F0Y, F0Z, EX, EY, EZ, PX, PY, PZ, P0X, P0Y, P0Z, RBX, RBY, RBZ, RTX, RTY, RTZ, RGX, RGY, RGZ, KX, KY, KZ, SBX, SBY, SBZ, STX, STY, STZ, S2X, S2Y or S2Z, (Real>0.0 or Blank). Default=SYMV3. SYMV4 requires SYMV3 to be non blank.
2	STHICK	Thickness of the stampable sheet(s). Required for SBX, SBY, SBZ, STX, STY, STZ, S2X, S2Y, S2Z. Optional for FBX, FBY, FBZ, FTX, FTY, FTZ, FGX, FGY, FGZ. (Real > 0 or -1.0≤Real<0). A positive value enters the exact value of the thickness. A negative value will cause the program to calculate the thickness as a fraction of the local height. The fraction is the absolute value of STHICK.
3	PUNCH	Sheet/Casting integrity. Optionally used with SBX, SBY, SBZ, STX, STY, STZ, S2X, S2Y, S2Z, FBX, FBY, FBZ, FTX, FTY, FTZ, FGX, FGY, FGZ. ("YES" or "NO" or Blank). YES will allow holes though the sheet. NO will not allow holes through the sheet. Default is YES.
4	VOIDDNS	Density of the void area. Optionally used with SBX, SBY, SBZ, STX, STY, STZ, S2X, S2Y, S2Z (Real>0.0 or Blank) Default=0.001.
5	OFFSET	Starting location of the sheet. Optionally used with SBX, SBY, SBZ, STX, STY, STZ, S2X, S2Y, S2Z (0≤Real≤1.0 or Blank). OFFSET is a fraction of the local height away from the top and/or bottom. Default=0.0.

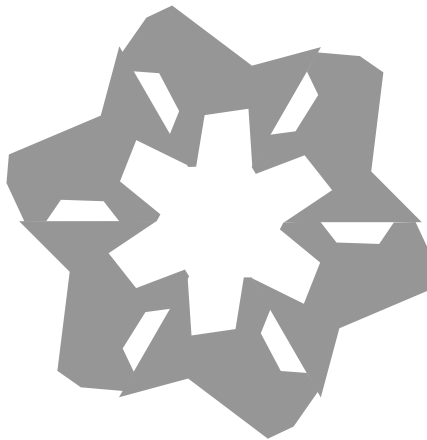
Remarks:

1. This entry has to be selected by **TPROP** data to be used.
2. The first two continuation lines must be provided. The third continuation line must be provided if any TYPEi is FBX, FBY, FBZ, FTX, FTY, FTZ, FGX, FGY, FGZ, F0X, F0Y, F0Z, FSX, FSY, FSZ, RBX, RBY, RBZ, RTX, RTY, RTZ, RGX, RGY, RGZ, KX, KY or KZ. The third and fourth continuation lines must be provided if any TYPEi is SBX, SBY, SBZ, STX, STY, STZ, S2X, S2Y or S2Z.
3. The three points (A1, A2, A3), (B1, B2, B3) and (C1, C2, C3) define three directions X, Y and Z. The three points must be unique and non-colinear.

- An RID of zero references the basic coordinate system.



- If CX, CY or CZ are used, then field n must be given, and n cyclically symmetric sections are created about specified axis. For example if $n = 6$, a solution might look like:



- Up to three fabrication constraints are allowed per entry. Following is a table with available fabrication constraints. See [Filling Types](#) (p. 390) for an illustration of the filling constraint types.

TYPE	Description of Fabrication Constraints
MXY	Mirror symmetry with respect to the XY plane
MYZ	Mirror symmetry with respect to the YZ plane
MZX	Mirror symmetry with respect to the ZX plane
CX	Cyclic symmetry about the X axis
CY	Cyclic symmetry about the Y axis
CZ	Cyclic symmetry about the Z axis

EX	Extrusion along the X axis
EY	Extrusion along the Y axis
EZ	Extrusion along the Z axis
UXY	Uniform in planes parallel to the XY plane
UYZ	Uniform in planes parallel to the YZ pane
UZX	Uniform in planes parallel to the ZX plane
UXYZ	Uniform in all directions
PX	Periodic pattern repetition in the X direction (with user specified pitch pattern distance DISTX)
PY	Periodic pattern repetition in the Y direction (with user specified pitch pattern distance DISTY)
PZ	Periodic pattern repetition in the Z direction (with user specified pitch pattern distance DISTZ)
P0X	Symmetric Periodic pattern repetition in the X and -X directions (with user specified pitch pattern distance DISTX)
P0Y	Symmetric Periodic pattern repetition in the Y and -Y directions (with user specified pitch pattern distance DISTY)
P0Z	Symmetric Periodic pattern repetition in the Z and -Z directions (with user specified pitch pattern distance DISTZ)
FBX	Filling in the +X direction
FBY	Filling in the +Y direction
FBZ	Filling in the +Z direction
FTX	Filling in the -X direction
FTY	Filling in the -Y direction
FTZ	Filling in the -Z direction
FSX	Filling simulatneously from +X and -X directions (filling from the outside in)
FSY	Filling simulatneously from +Y and -Y directions (filling from the outside in)
FSZ	Filling simulatneously from +Z and -Z directions (filling from the outside in)

FGX	Filling on +X and -X directions from general surface (filling from the inside out)
FGY	Filling on +Y and -Y directions from general surface (filling from the inside out)
FGZ	Filling on +Z and -Z directions from general surface (filling from the inside out)
F0X	Filling symmetrically from X=0.0 toward the top and bottom (filling from the inside out)
F0Y	Filling symmetrically from Y=0.0 toward the top and bottom (filling from the inside out)
F0Z	Filling symmetrically from Z=0.0 toward the top and bottom (filling from the inside out)
RBX	Radial Filling From the Inner Surface about the X axis
RBY	Radial Filling From the Inner Surface about the Y axis
RBZ	Radial Filling From the Inner Surface about the Z axis
RTX	Radial Filling From the Outer Surface about the X axis
RTY	Radial Filling From the Outer Surface about the Y axis
RTZ	Radial Filling From the Outer Surface about the Z axis
RGX	Radial Filling From the General Surface about the X axis (from the interior to the inner and outer surfaces)
RGY	Radial Filling From the General Surface about the Y axis (from the interior to the inner and outer surfaces)
RGZ	Radial Filling From the General Surface about the Z axis (from the interior to the inner and outer surfaces)
KX	Radial Spokes about the X axis
KY	Radial Spokes about the Y axis
KZ	Radial Spokes about the Z axis
SBX	Sheet forming normal to +X (1 Layer) (initial configuration starts at the bottom)
SBY	Sheet forming normal to +Y (1 Layer) (initial configuration starts at the bottom)

SBZ	Sheet forming normal to +Z (1 Layer) (initial configuration starts at the bottom)
STX	Sheet forming normal to +X (1 Layer) (initial configuration starts at the top)
STY	Sheet forming normal to +Y (1 Layer) (initial configuration starts at the top)
STZ	Sheet forming normal to +Z (1 Layer) (initial configuration starts at the top)
S2X	Sheet forming normal to +X (2 Layer) (initial configuration of first layer starts at the top; initial configuration of second layer starts at the bottom)
S2Y	Sheet forming normal to +Y (2 Layer) (initial configuration of first layer starts at the top; initial configuration of second layer starts at the bottom)
S2Z	Sheet forming normal to +Z (2 Layer) (initial configuration of first layer starts at the top; initial configuration of second layer starts at the bottom)

- Two types of symmetries method are used internally in the program: Element based symmetry or location based symmetries. Element based symmetries enforce symmetries using symmetry of the elements in the mesh. Location based symmetry enforce symmetry based on symmetry of the space. Location based symmetries requires the SYMV1 field be specified, and are currently only available for extrusion, filling and stamping. For location based symmetries, the minimum member size is approximately $\text{SYMV1} + 2 \cdot \text{SYMV2}$.
- Not all fabrication constraints can be mixed together. Following is a table with the allowed combinations (excluding periodic):

MX+MYZ, MX+MZX, MYZ+MZX, MX+MYZ+MZX
CX+MYZ, CY+MZX, CZ+MX
EX+MX, EX+MZX, EX+MX+MZX EY+MYZ, EY+MX, EY+MYZ+MX EZ+MZX, EZ+MYZ, EZ+MZX+MYZ
EX+CX, EY+CY, EZ+CZ
UYZ+MYZ, UZX+MZX, UXY+MX
FiX+MX, FiX+MZX, FiX+MX+MZX FiY+MYZ, FiY+MX, FiY+MYZ+MX FiZ+MZX, FiZ+MYZ, FiZ+MZX+MYZ
FiX+CX, FiY+CY, FiZ+CZ

$RiX+MXY, RiX+MYZ, RiX+ MZX,$ $RiX+MXY+MYZ,RiX+MXY+MZX,RiX+MYZ+MZX$ $RiY+MXY, RiY+MYZ, RiY+ MZX,$ $RiY+MXY+MYZ,RiY+MXY+MZX,RiY+MYZ+MZX$ $RiZ+MXY, RiZ+MYZ, RiZ+ MZX,$ $RiZ+MXY+MYZ,RiZ+MXY+MZX,RiZ+MYZ+MZX$
$RiX+CX, RiY+CY, RiZ+CZ$
$RiX+EX, RiY+EY, RiZ+EZ$
$RiX+CX+EX, RiY+CY+EY, RiZ+CZ+EZ$
$RiX+CX+MYZ, RiY+CY+MZX, RiZ+CZ+MXZ$
$RiX+EX+MXY, RiX+EX+MZX, RiY+EY+MXY,RiY+EY+MYZ, RiZ+EZ+MYZ,$ $RiZ+EZ+MZX,$

9. In the above table:

FiX is one of $FBX, FTX, FSX, FGX, F0X, SBX, STX$ or $S2X$

FiY is one of $FBY, FTY, FSY, FGY, F0Y, SBY, STY$ or $S2Y$

FiZ is one of $FBZ, FTZ, FSZ, FGZ, F0Z, SBZ, STZ$ or $S2Z$

RiX is one of $RBX, RTX, RGX,$ or KX

RiY is one of $RBY, RTY, RGY,$ or KY

RiZ is one of $RBZ, RTZ, RGZ,$ or KZ

10. Periodic structures with repetetions on the x directions requires using PX (or $P0X$) and $DISTX$. Periodic structures with repetetions on the y directions requires using PY (or $P0Y$) and $DISTY$. Periodic structures with repetetions on the z directions requires using PZ (or $P0Z$) and $DISTZ$.

11. Periodic fabrication constraints and their valids combination with other fabrication constraints are:

Types that can combined with Periodic	Combinations	Number of combinations
2 or 3 Periodic	$PX+PY, PX+PZ, PY+PZ, PX+PY+PZ$	4
Periodic + 1 or 2 Mirrors	$PX+MXY, PX+MZX, PX+MXY+MZX,$ $PY+MXY, PY+MYZ, PY+MXY+MYZ,$ $PZ+MZX, PZ+MYZ, PZ+MZX+MYZ$	9
2 Periodic + Mirror	$PX+PY +MXY,$ $PX+PZ +MZX,$ $PY+PZ +MYZ$	3
Periodic + Cyclic	$PX+CX, PY+CY, PZ+CZ$	3

Periodic + Extrusion	PX+EY, PX+EZ, PY+EX, PY+EZ, PZ+EX, PZ+EY	6
2 Periodic + Extrusion	PX+PY+EZ, PX+PZ+EY, PY+PZ+EX	3
Periodic + Mirror + Extrusion	PX+MXY+EY, PX+MZX+EZ, PY+MXY+EX, PY+MYZ+EZ, PZ+MZX+EX, PZ+MYZ+EY	6
Periodic + Uniform	PX+UYZ, PY+UZX, PZ+UXY	3
Periodic + Filling	PX+FiY, PX+FiZ, PY+FiX, PY+FiZ, PZ+FiX, PZ+FiY	6
2 Periodic + Filling	PX+PY+FiZ, PX+PZ+FiY, PY+PZ+FiX,	3
Periodic + Mirror + Filling	PX+MXY+FiY, PX+MZX+FiZ, PY+MXY+FiX, PY+MYZ+FiZ, PZ+MZX+FiX, PZ+MYZ+FiY	6
Periodic + Radial	RiX+PX, RiY+PY, RiZ+ PZ	3
Periodic + Mirror + Radial	PX+MXY+RiX, PX+MZX+RiX, PY+MXY+RiY, PY+MYZ+RiY, PZ+MYZ+RiZ, PZ+MZX+RiZ	6
Periodic +Cyclic + Radial	PX+CX+RiX PY+CY+RiY, PZ+CZ+RiZ	3

12. In the above table:

FiX is one of FBX, FTX, FSX, FGX, F0X, SBX, STX or S2X

FiY is one of FBY, FTY, FSY, FGY, F0Y, SBY, STY or S2Y

FiZ is one of FBZ, FTZ, FSZ, FGZ, F0Z, SBZ, STZ or S2Z

RiX is one of RBX, RTX, RGX, or KX

RiY is one of RBY, RTY, RGY, or KY

RiZ is one of RBZ, RTZ, RGZ, or KZ

PX is one of PX or P0X

PY is one of PY or P0Y

PZ is one of PZ or P0Z

13. Minimum member size can be obtained without the need of symmetries, the minimum member size is approximately SYMV1 + 2*SYMV2.

14. UXYZ can not be mixed with any other fabrication constraints.

9.2.19 TSYM3

Data Entry: **TSYM3** - Fabrication Constraints for Topology Optimization, Form 3.

Description: Define fabrication constraints on a coordinate system defined by CORDxx.

Format:

1	2	3	4	5	6	7	8	9	10
TSYM3	ID	CID	n	NSECT	DISTX	DISTY	DISTZ		
+	TYPE1	TYPE2	TYPE3						
+	SYMV1	SYMV2	SYMV3	SYMV4					
+	STHICK	PUNCH	VOIDDNS	OFFSET					

Example 1: Filling in the +x direction, Minimum member size = 5.0 mm. Plane of symmetry defined by CORDxx = 130:

1	2	3	4	5	6	7	8	9	10
TSYM3	301	1301							
+	FX								
+	5.0								

Example 2: Triple Symmetry Conditions. Plane of symmetry defined by CORDxx = 1302.

1	2	3	4	5	6	7	8	9	10
TSYM3	302	1302							
+	MYZ	MZX	MXZ						

Example 4: Two stampable layers on x direction with mirror symmetries on ZX and YZ planes. Plane of symmetry defined by CORDxx = 1303 Each layer should be 2.00 mm thick with not inner holes, void density=0.075:.

1	2	3	4	5	6	7	8	9	10
TSYM3	303	1303							
+	S2X	MZX	MYZ						
+	5.0								
+	2.0	NO	0.07						

Field Information Description

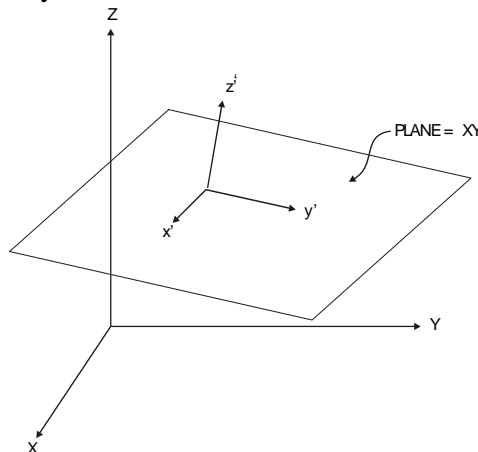
2	ID	Unique TSYM identification number. (Integer > 0).
3	CID	Coordinate system identification number. (Integer ≥ 0 or Blank).
4	n	Number of cyclic symmetry sections (Integer > 1 or Blank).

5	NSECT	Number of sectors. Used with RBX, RBY, RBZ, RTX, RTY, RTZ, RGX, RGY, RGZ, KX, KY or KZ (Integer > 1 or Blank). Default = 36 or 6*n when n is not zero.
6,7,8	DISTX,DISTY, DISTZ	Pattern pitch distance X (or Y or Z). (Real > 0 or blank). A positive value will cause the software to generate a design proposal which features are periodic in the X (or Y or Z) direction. The pattern pitch is the distance where the patterns repeats. Pattern pitches are used together with PX, PY, PZ, P0X, P0Y and/or P0Z. See Remarks 9, 10 & 11.
2	TYPE1	Fabrication constraint type. One of "MYZ", "MZX", "MXY", "CX", "CY", "CZ", "EX", "EY", "EZ", "FBX", "FBY", "FBZ", "FTX", "FTY", "FTZ", "FSX", "FSY", "FSZ", "FGX", "FGY", "FGZ", "F0X", "F0Y", "F0Z", "PX", "PY", "PZ", "P0X", "P0Y", "P0Z", "RBX", "RBY", "RBZ", "RTX", "RTY", "RTZ", "RGX", "RGY", "RGZ", "KX", "KY" or "KZ", "SBX", "SBY", "SBZ", "STX", "STY", "STZ", "S2X", "S2Y", "S2Z", "UYZ", "UZX", "UXY" or "UXYZ",
3,4	TYPE2, TYPE3	Fabrication constraint type. One of "MYZ", "MZX", "MXY", "CX", "CY", "CZ", "EX", "EY", "EZ", "FBX", "FBY", "FBZ", "FTX", "FTY", "FTZ", "FSX", "FSY", "FSZ", "FGX", "FGY", "FGZ", "F0X", "F0Y", "F0Z", "PX", "PY", "PZ", "P0X", "P0Y", "P0Z", "RGX", "RGY", "RGZ", "KX", "KY" or "KZ", "SBX", "SBY", "SBZ", "STX", "STY", "STZ", "S2X", "S2Y", "S2Z", "UYZ", "UZX", "UXY" or "UXYZ" or Blank. See Remarks 6 and 7.
2	SYMV1	Pole spacing. Required for FBX, FBY, FBZ, FTX, FTY, FTZ, FSX, FSY, FSZ, FGX, FGY, FGZ, F0X, F0Y, F0Z, RBX, RBY, RBZ, RTX, RTY, RTZ, RGX, RGY, RGZ, KX, KY, KZ, SBX, SBY, SBZ, STX, STY, STZ, S2X, S2Y or S2Z, (Real>0.0). Optional for EX, EY, EZ, PX, PY, PZ, P0X, P0Y, or P0Z.
3	SYMV2	Spread size. Optionally used with FBX, FBY, FBZ, FTX, FTY, FTZ, FSX, FSY, FSZ, FGX, FGY, FGZ, F0X, F0Y, F0Z, EX, EY, EZ, PX, PY, PZ, P0X, P0Y, P0Z, RBX, RBY, RBZ, RTX, RTY, RTZ, RGX, RGY, RGZ, KX, KY or KZ, SBX, SBY, SBZ, STX, STY, STZ, S2X, S2Y or S2Z, (Real≥0.0 or Blank). Default=0.0.
3	SYMV3	Maximum member size. Optionally used with FBX, FBY, FBZ, FTX, FTY, FTZ, FSX, FSY, FSZ, FGX, FGY, FGZ, F0X, F0Y, F0Z, EX, EY or EZ, PX, PY, PZ, P0X, P0Y, P0Z, RBX, RBY, RBZ, RTX, RTY, RTZ, RGX, RGY, RGZ, KX, KY or KZ, SBX, SBY, SBZ, STX, STY, STZ, S2X, S2Y or S2Z, (Real>0.0). SYMV3 requires SYMV1 to be non blank.
4	SYMV4	Minimum Gap. Optionally used with FBX, FBY, FBZ, FTX, FTY, FTZ, FSX, FSY, FSZ, FGX, FGY, FGZ, F0X, F0Y, F0Z, EX, EY, EZ, PX, PY, PZ, P0X, P0Y, P0Z, RBX, RBY, RBZ, RTX, RTY, RTZ, RGX, RGY, RGZ, KX, KY, KZ, SBX, SBY, SBZ, STX, STY, STZ, S2X, S2Y or S2Z, (Real>0.0 or Blank). Default=SYMV3. SYMV4 requires SYMV3 to be non blank.

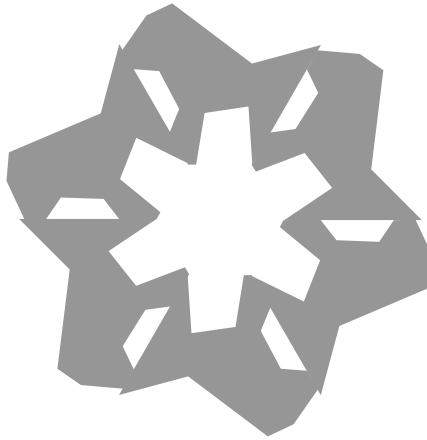
2	STHICK	<p>Thickness of the stampable sheet(s). Required for SBX, SBY, SBZ, STX, STY, STZ, S2X, S2Y, S2Z. Optional for FBX, FBY, FBZ, FTX, FTY, FTZ, FGX, FGY, FGZ.</p> <p>(Real > 0 or $-1.0 \leq \text{Real} < 0$).</p> <p>A positive value enters the exact value of the thickness.</p> <p>A negative value will cause the program to calculate the thickness as a fraction of the local height. The fraction is the absolute value of STHICK.</p>
3	PUNCH	<p>Sheet/Casting integrity. Optionally used with SBX, SBY, SBZ, STX, STY, STZ, S2X, S2Y, S2Z, FBX, FBY, FBZ, FTX, FTY, FTZ, FGX, FGY, FGZ.</p> <p>("YES" or "NO" or Blank). YES will allow holes through the sheet. NO will not allow holes through the sheet. Default is YES.</p>
4	VOIDDNS	<p>Density of the void area. Optionally used with SBX, SBY, SBZ, STX, STY, STZ, S2X, S2Y, S2Z (Real>0.0 or Blank)</p> <p>Default=0.001.</p>
5	OFFSET	<p>Starting location of the sheet. Optionally used with SBX, SBY, SBZ, STX, STY, STZ, S2X, S2Y, S2Z ($0 \leq \text{Real} \leq 1.0$ or Blank).</p> <p>OFFSET is a fraction of the local height away from the top and/or bottom. Default=0.0.</p>

Remarks:

1. This entry has to be selected by **TPROP** data to be used.
2. The first continuation line must be provided. The second continuation line must be provided if any TYPEi is FBX, FBY, FBZ, FTX, FTY, FTZ, FGX, FGY, FGZ, FOX, FOY, FOZ, FSX, FSY, FSZ, RBX, RBY, RBZ, RTX, RTY, RTZ, RGX, RGY, RGZ, KX, KY or KZ. The second and third continuation lines must be provided if any TYPEi is SBX, SBY, SBZ, STX, STY, STZ, S2X, S2Y or S2Z.
3. The coordinate system defines three directions, X, Y, and Z.



4. If CX, CY or CZ are used, then field n must be given, and n cyclically symmetric sections are created about specified axis. For example if $n = 6$, a solution might look like:



5. Up to three fabrication constraints are allowed per entry. Following is a table with available fabrication constraints. See [Filling Types](#) (p. 390) for an illustration of the filling constraint types.

TYPE	Description of Fabrication Constraints
MXY	Mirror symmetry with respect to the XY plane
MYZ	Mirror symmetry with respect to the YZ pane
MZX	Mirror symmetry with respect to the ZX plane
CX	Cyclic symmetry about the X axis
CY	Cyclic symmetry about the Y axis
CZ	Cyclic symmetry about the Z axis
EX	Extrusion along the X axis
EY	Extrusion along the Y axis
EZ	Extrusion along the Z axis
UXY	Uniform in planes parallel to the XY plane
UYZ	Uniform in planes parallel to the YZ pane
UZX	Uniform in planes parallel to the ZX plane
UXYZ	Uniform in all directions
PX	Periodic pattern repetition in the X direction (with user specified pitch pattern distance DISTX)

PY	Periodic pattern repetition in the Y direction (with user specified pitch pattern distance DISTY)
PZ	Periodic pattern repetition in the Z direction (with user specified pitch pattern distance DISTZ)
P0X	Symmetric Periodic pattern repetition in the X and -X directions (with user specified pitch pattern distance DISTX)
P0Y	Symmetric Periodic pattern repetition in the Y and -Y directions (with user specified pitch pattern distance DISTY)
P0Z	Symmetric Periodic pattern repetition in the Z and -Z directions (with user specified pitch pattern distance DISTZ)
FBX	Filling in the +X direction
FBY	Filling in the +Y direction
FBZ	Filling in the +Z direction
FTX	Filling in the -X direction
FTY	Filling in the -Y direction
FTZ	Filling in the -Z direction
FSX	Filling simultaneously from +X and -X directions (filling from the outside in)
FSY	Filling simultaneously from +Y and -Y directions (filling from the outside in)
FSZ	Filling simultaneously from +Z and -Z directions (filling from the outside in)
FGX	Filling on +X and -X directions from general surface (filling from the inside out)
FGY	Filling on +Y and -Y directions from general surface (filling from the inside out)
FGZ	Filling on +Z and -Z directions from general surface (filling from the inside out)
F0X	Filling symmetrically from X=0.0 toward the top and bottom (filling from the inside out)
F0Y	Filling symmetrically from Y=0.0 toward the top and bottom (filling from the inside out)

F0Z	Filling symmetrically from Z=0.0 toward the top and bottom (filling from the inside out)
RBX	Radial Filling From the Inner Surface about the X axis
RBY	Radial Filling From the Inner Surface about the Y axis
RBZ	Radial Filling From the Inner Surface about the Z axis
RTX	Radial Filling From the Outer Surface about the X axis
RTY	Radial Filling From the Outer Surface about the Y axis
RTZ	Radial Filling From the Outer Surface about the Z axis
RGX	Radial Filling From the General Surface about the X axis (from the interior to the inner and outer surfaces)
RGY	Radial Filling From the General Surface about the Y axis (from the interior to the inner and outer surfaces)
RGZ	Radial Filling From the General Surface about the Z axis (from the interior to the inner and outer surfaces)
KX	Radial Spokes about the X axis
KY	Radial Spokes about the Y axis
KZ	Radial Spokes about the Z axis
SBX	Sheet forming normal to +X (1 Layer) (initial configuration starts at the bottom)
SBY	Sheet forming normal to +Y (1 Layer) (initial configuration starts at the bottom)
SBZ	Sheet forming normal to +Z (1 Layer) (initial configuration starts at the bottom)
STX	Sheet forming normal to +X (1 Layer) (initial configuration starts at the top)
STY	Sheet forming normal to +Y (1 Layer) (initial configuration starts at the top)
STZ	Sheet forming normal to +Z (1 Layer) (initial configuration starts at the top)
S2X	Sheet forming normal to +X (2 Layer) (initial configuration of first layer starts at the top; initial configuration of second layer starts at the bottom)

S2Y	Sheet forming normal to +Y (2 Layer) (initial configuration of first layer starts at the top; initial configuration of second layer starts at the bottom)
S2Z	Sheet forming normal to +Z (2 Layer) (initial configuration of first layer starts at the top; initial configuration of second layer starts at the bottom)

- Two types of symmetries method are used internally in the program: Element based symmetry or location based symmetries. Element based symmetries enforce symmetries using symmetry of the elements on the mesh. Location based symmetry enforce symmetry based on symmetry of the space. Location based symmetries requires the SYMV1 field be specified, and are currently only available for extrusion, filling and stamping. For location based symmetries, the minimum member size is approximately SYMV1 + 2*SYMV2.
- Not all fabrication constraints can be mixed together. Following is a table with the allowed combinations (excluding periodic):

MX+MYZ, MX+MZX, MYZ+MZX, MX+MYZ+MZX
CX+MYZ, CY+MZX, CZ+MX
EX+MX, EX+MZX, EX+MX+MZX EY+MYZ, EY+MX, EY+MYZ+MX EZ+MZX, EZ+MYZ, EZ+MZX+MYZ
EX+CX, EY+CY, EZ+CZ
UYZ+MYZ, UZX+MZX, UXY+MX
FiX+MX, FiX+MZX, FiX+MX+MZX FiY+MYZ, FiY+MX, FiY+MYZ+MX FiZ+MZX, FiZ+MYZ, FiZ+MZX+MYZ
FiX+CX, FiY+CY, FiZ+CZ
RiX+MX, RiX+MYZ, RiX+ MZX, RiX+MX+MYZ,RiX+MX+MZX,RiX+MYZ+MZX RiY+MX, RiY+MYZ, RiY+ MZX, RiY+MX+MYZ,RiY+MX+MZX,RiY+MYZ+MZX RiZ+MX, RiZ+MYZ, RiZ+ MZX, RiZ+MX+MYZ,RiZ+MX+MZX,RiZ+MYZ+MZX
RiX+CX, RiY+CY, RiZ+CZ
RiX+EX, RiY+EY, RiZ+EZ
RiX+CX+EX, RiY+CY+EY, RiZ+CZ+EZ
RiX+CX+MYZ, RiY+CY+MZX, RiZ+CZ+MX
RiX+EX+MX, RiX+EX+MZX, RiY+EY+MX,RiY+EY+MYZ, RiZ+EZ+MYZ, RiZ+EZ+MZX,

- In the above table:

FiX is one of FBX, FTX, FSX, FGX, F0X, SBX, STX or S2X

FiY is one of FBY, FTY, FSY, FGY, F0Y, SBY, STY or S2Y

FiZ is one of FBZ, FTZ, FSZ, FGZ, F0Z, SBZ, STZ or S2Z

RiX is one of RBX, RTX, RGX, or KX

RiY is one of RBY, RTY, RGY, or KY

RiZ is one of RBZ, RTZ, RGZ, or KZ

9. Periodic structures with repetitions on the x directions requires using PX (or P0X) and DISTX. Periodic structures with repetitions on the y directions requires using PY (or P0Y) and DISTY. Periodic structures with repetitions on the z directions requires using PZ (or P0Z) and DISTZ.
10. Periodic fabrication constraints and their valids combination with other fabrication constraints are:

Types that can combined with Periodic	Combinations	Number of combinations
2 or 3 Periodic	PX+PY, PX+PZ, PY+PZ, PX+PY+PZ	4
Periodic + 1 or 2 Mirrors	PX+MX, PX+MZ, PX+MX+MZ, PY+MX, PY+MY, PY+MX+MY, PZ+MX, PZ+MY, PZ+MX+MY	9
2 Periodic + Mirror	PX+PY +MX, PX+PZ +MZ, PY+PZ +MY	3
Periodic + Cyclic	PX+CX, PY+CY, PZ+CZ	3
Periodic + Extrusion	PX+EY, PX+EZ, PY+EX, PY+EZ, PZ+EX, PZ+EY	6
2 Periodic + Extrusion	PX+PY+EZ, PX+PZ+EY, PY+PZ+EX	3
Periodic + Mirror + Extrusion	PX+MX+EY, PX+MZ+EZ, PY+MX+EX, PY+MY+EZ, PZ+MX+EX, PZ+MY+EY	6
Periodic + Uniform	PX+UY, PY+UZ, PZ+UX	3
Periodic + Filling	PX+FiX, PX+FiZ, PY+FiX, PY+FiZ, PZ+FiX, PZ+FiY	6
2 Periodic + Filling	PX+PY+FiZ, PX+PZ+FiY, PY+PZ+FiX,	3

Periodic +Mirror + Filling	PX+MX+FiY, PX+MZ+FiZ, PY+MX+FiX,PY+MY+FiZ, PZ+MZ+FiX, PZ+MY+FiY	6
Periodic + Radial	RiX+PX, RiY+PY, RiZ+ PZ	3
Periodic + Mirror + Radial	PX+MX+RiX,PX+MZ+RiX, PY+MX+RiY,PY+MY+RiY, PZ+MY+RiZ,PZ+MZ+RiZ	6
Periodic +Cyclic + Radial	PX+CX+RiX PY+CY+RiY, PZ+CZ+RiZ	3

11. In the above table:

FiX is one of FBX, FTX, FSX, FGX, F0X, SBX, STX or S2X

FiY is one of FBY, FTY, FSY, FGY, F0Y, SBY, STY or S2Y

FiZ is one of FBZ, FTZ , FSZ, FGZ, F0Z, SBZ, STZ or S2Z

RiX is one of RBX, RTX, RGX, or KX

RiY is one of RBY, RTY, RGY,or KY

RiZ is one of RBZ, RTZ, RGZ, or KZ

PX is one of PX or P0X

PY is one of PY or P0Y

PZ is one of PZ or P0Z

12. Minimum member size can be obtained without the need of symmetries, the minimum member size is approximately $\text{SYMV1} + 2 \cdot \text{SYMV2}$.

13. UXYZ can not be mixed with any other fabrication constraints.

9.2.20 TVAR

Data Entry: **TVAR** - Topology Extra Variable.

Description: Define a topology extra variable that may be modified during design optimization

Format:

1	2	3	4	5	6	7	8	9	10
TVAR	ID	LABEL	INIT	LB	UB	DELX	DXMIN		

Example:

1	2	3	4	5	6	7	8	9	10
TVAR	1	ROD1	5.0	0.01	100.0	0.5	0.05		

Field Information Description

2	ID	Unique topology extra variable identification number (Integer > 0).
3	LABEL	User supplied name for printing purposes (or blank).
4	INIT	Initial value. (Real).
5	LB	Lower bound. (Real. $LB \leq INIT$).
6	UB	Upper bound. (Real. $UB \geq INIT$).
7	DELX	Design variable fractional move limit (Real > 0 or blank). See remark 1.
8	DXMIN	Design variable minimum move limit factor (Real > 0 or blank). See remark 1.

Remarks:

1. DELX and DXMIN default to 0.5 and 0.1 respectively, unless these values are specified on the DOPT entry (in which case they default to the values specified on DOPT). The minimum move limit values is defined as:

$$DXMIN = \begin{cases} DXMIN & \text{if } |DV_{\text{initial}}| \leq 1.0 \\ DXMIN|DV_{\text{initial}}| & \text{if } |DV_{\text{initial}}| > 1.0 \end{cases}$$

2. The initial value may be changed from INIT to LB or UB depending on the setting of the DOPT parameter DVINIT. The default is not to change the initial value.
3. Topology extra variables cannot be directly associated with any parameter values in the model. They may only be included in the topology optimization problem by referencing them with **TRESP2** or **TRESP3** entries.

CHAPTER 10

Multi-Model Master Data

- Multi-Model Optimization
- Master Data Executive Control
- Master Data Solution Control
- Master Data Bulk Data

10.1 Multi-Model Optimization

GENESIS includes a mode that allows multiple optimization models to run simultaneously via a master optimization process. The individual sub-models may share some or all of their design variables with other sub-models. Each sub-model may include its own private constraints and objectives. Using multi-model optimization, large-scale optimization problems may be solved on HPC clusters, with each sub-model analysis solver running on a separate compute node.

The primary benefits of multi-model optimization are:

1. The ability to optimize large-scale models where designable regions are common to several model assemblies, but remaining parts of the models differ. For example, optimization of components common to several configurations of a vehicle, such as sedan and convertible versions.
2. Reducing the wall clock time of large scale problems by splitting the solution of different boundary conditions onto separate compute nodes.

To run multi-models optimization, each sub-model input data should be prepared as if it were a standard single model problem. Linkages among sub-model design variables are created automatically by common DVAR ID, common TVAR ID and by common properties (same property ID and identical element meshes) that are designed by TPROP, DSPLIT or DTGRID.

An additional input data, the Multi-Model Optimization Master Data file, must also be prepared. At a minimum, this master data will define the input data files of the individual sub-models. In addition, the master data must define selected input data that must be common to all sub-models. This common data from the master file will override any settings of the equivalent data in the sub-models. Finally, the master data may also define synthetic responses that combine response values from multiple sub-models, and set additional objectives or constraints using these synthetic responses.

10.2 Master Data Executive Control

The executive control portion of the input data is used to control the overall program flow, data checking, and diagnostic printing. At least one **MODEL** entry is required. All other data is optional.

Only the first four characters of each keyword need be used.

Comments are allowed and are indicated by a “\$” as the first character.

10.2.1 \$

Executive Control Entry: **\$** - Comment

Description: Enter a comment line.

Format:

\$ Any character data

Example:

\$ This line is a comment.

10.2.2 ANALYSIS

Executive Control Entry: **ANALYSIS** - Program Flow Control

Description: Stops the program after the first analysis.

Format:

ANALYSIS

Example:

ANALYSIS

Remarks:

1. The following executive control commands are mutually exclusive (only one may be specified): ANALYSIS, **CHECK**, **SENSITIVITY**.
2. *GENESIS* will perform the analysis after updating the model. In other words, the design data will be used to update grid and property data if there is shape/sizing data, and/or update material data if there is topology data.
3. This command will be imposed on all sub-models. Any ANALYSIS, CHECK or SENSITIVITY command in a sub-model data file will be ignored.

10.2.3 CEND

Executive Control Entry: **CEND** - Mark the End of Executive Control

Description: Delimits the Executive Control section of the input file from the Solution Control section.

Format:

CEND

Example:

CEND

Remarks:

1. If the CEND command is not present in the input, the Solution Control will begin at the line containing the first detected Solution Control command.

10.2.4 CHECK

Executive Control Entry: **CHECK** - Program Flow Control

Description: Stops the program after checking the input data.

Format:

CHECK

Example:

CHECK

Remarks:

1. The following executive control commands are mutually exclusive (only one may be specified): **ANALYSIS**, **CHECK**, **SENSITIVITY**
2. This can be used to save time when large amounts of input data are being assembled.
3. This command will be imposed on all sub-models. Any **ANALYSIS**, **CHECK** or **SENSITIVITY** command in a sub-model data file will be ignored.

10.2.5 DIAG

Executive Control Entry: **DIAG** - Program Diagnostic Control

Description: Enables diagnostic printing or alternate algorithms.

Format:

DIAG = n1,n2,n3,...

Example:

DIAG=87

Option	Meaning
ni	Diagnostic value to set(Integer>0).

Remarks:

1. See Volume 1 (Analysis Reference) for a description of the information printed for supported values of “ni”.
2. Note that the use of undocumented values of “ni” may produce unexpected results.
3. Multiple DIAG commands are allowed.

10.2.6 DRESP3

Executive Control Entry: **DRESP3** - External Response Control

Description: Defines the name of an external shared object (DLL) that will calculate external user-defined responses for the DRESP3 or TRESP3 bulk data entry.

Format:

DRESP3 = *lid*, full_path_to_shared_object

Alternate Format:

DRESP3 = full_path_to_shared_object

Examples:

DRESP3 = 2, /home/mrt/work/dresp3.so

DRESP3 = 17, D:\users\mrt\dresp3.dll

Option	Meaning
--------	---------

<i>lid</i>	Unique DRESP3 library identification number (Integer >0).
------------	---

Remarks:

1. If no library ID is specified, the library will be used to handle all IDs not specifically defined. At most one DRESP3 command without a library ID is allowed.
2. This command may not be available on all operating systems.
3. **DRESP3** or **TRESP3** bulk data entries must exist to request external user-defined responses.
4. The shared object must export one required interface function. The interface function has the following Fortran declaration:

```
SUBROUTINE DRESP3(IWHICH, VAR, N, VAL, IERR)
  INTEGER IWHICH, N, IERR
  DOUBLE PRECISION VAR(*), VAL
```

Note that if a language other than Fortran is used to create the shared object, care must be taken to ensure that the correct interface function name is exported. The actual required function name is system dependent. For example, using the C language, the function should be named as follows:

Microsoft Windows	DRESP3
Solaris, Linux, HP-UX	dresp3_
AIX	dresp3

For more details see [Use of the DRESP3 / TRESP3 capability](#) (p. 307).

10.2.7 INCLUDE

Executive Control Entry: **INCLUDE**

Description: Select an external file that contains executive control statements.

Format:

```
INCLUDE 'file name'
```

Alternate Format:

```
INCLUDE = file name
```

Examples:

```
INCLUDE 'myscript.lua'
```

```
INCLUDE = K2UU_LIST.TXT
```

Option	Meaning
--------	---------

file name	External file name. The user must provide the file name according to the machine installation.
-----------	--

Remarks:

1. Multiple INCLUDE data are allowed in the executive control.
2. The external file cannot contain INCLUDE data statements.
3. The external file cannot contain the **CEND** delimiter.
4. The file name is limited to 240 characters.
5. If the quoted format is used, and the line does not end with a quote character, additional lines will be read until the closing quote is found. Leading and trailing spaces on continued and continuation lines are discarded.

10.2.8 IOBUFF

Executive Control Entry: **IOBUFF** - Input/Output Control

Description: Defines memory buffering characteristics for optimizer process scratch file Input/Output

Format:

$$\text{IOBUFF} = n, mK$$

Examples:

$$\text{IOBUFF} = 64, 256K$$

Option	Meaning
n	The number of Input/Output buffers to use (Integer > 1).
mK	m is the size of each buffer in kilowords (Integer > 7).

Remarks:

1. The use of I/O buffers will take memory in addition to that specified by the **LENVEC** entry. The total amount of additional memory used will be $n*m$ kilowords (1 word = 4 bytes).
2. The minimum allowable buffersize is 8 kilowords.
3. If invalid values are specified, they will be reset to the closest acceptable values.

10.2.9 LENVEC

Executive Control Entry: **LENVEC** - Memory Control

Description: Specifies the amount of memory the optimizer process should use.

Format:

$$\text{LENVEC} = n$$

Alternate Format:

$$\text{LENVEC} = mK$$

Alternate Format:

$$\text{LENVEC} = mM$$

Alternate Format:

$$\text{LENVEC} = mG$$

Examples:

$$\text{LENVEC} = 85000000$$

$$\text{LENVEC} = 1500000K$$

$$\text{LENVEC} = 1500M$$

Option	Meaning
<i>n</i>	The number of words for the main storage array in <i>GENESIS</i> . The default value of n is installation dependent. On most systems, one word is 4 bytes. (1,000,000,000>Integer>0).
<i>mK</i>	m is the number of words in thousands for the main storage array in <i>GENESIS</i> . (Integer>0).
<i>mM</i>	m is the number of words in millions for the main storage array in <i>GENESIS</i> . (Integer>0).
<i>mG</i>	m is the number of words in billions for the main storage array in <i>GENESIS</i> . (Integer>0).

10.2.10 MODEL

Executive Control Entry: **MODEL** - Model Definition Control

Description: Defines the name of a sub-model input data file.

Format:

MODEL = *mid*, path_to_input_data

Alternate Format:

MODEL = path_to_input_data

Examples:

MODEL = 2, model_ex_YLoad.dat

MODEL = model_ex_ZLoad.dat

Option	Meaning
--------	---------

<i>mid</i>	Unique MODEL identification number (Integer >0).
------------	--

Remarks:

1. At least one MODEL is required.
2. The model ID is only required to reference a response calculated by the sub-model with a master data DRESP2, DRESP3, TRESP2 or TRESP3.

10.2.11 RESTART

Executive Control Entry: **RESTART** - Program Flow Control

Description: Uses results from a previous optimization run to reset the design variable initial values.

Format:

RESTART = n,m

Alternate Format:

RESTART = LAST,k

Examples:

RESTART = 7

RESTART = LAST, 5

Option	Meaning
n	The design cycle from a previous run to restart from (Integer>0).
m	The new maximum number of design cycles (Integer>n or blank). Default is the value specified on the DOPT bulk data entry.
LAST	Word indicating to restart from the last design cycle in the *.HIS file.
k	The maximum number of <u>additional</u> design cycles (Integer>0 or blank). Default is 10.

Remarks:

1. The value m is the maximum total number of design cycles, not the maximum number for this run. That is, m includes the design cycles from the previous run(s). Conversely, when the alternate format is used, the value k is the maximum design cycles for this run only.
2. See **Restart from any Previous Design Cycle** (p. 322) for a more in depth discussion of the restart capability available in *GENESIS*.
3. This command will be imposed on all sub-models. Any RESTART command in a sub-model data file will be ignored.

10.2.12 SCRIPT

Executive Control Entry: **SCRIPT** - Customization Control

Description: Defines a customization script to alter normal program behavior.

Format:

```
SCRIPT = engine, timeout
script lines
.
.
ENDSCRIPT
```

Example:

```
SCRIPT = LUA
genesis.hook.READ = function()
    genesis.diag[1] = 2 -- same as DIAG=12
end
ENDSCRIPT
```

Option	Meaning
<i>engine</i>	Script engine. Word 'LUA'.
<i>timeout</i>	Script timeout (in secs). If any script function takes longer than timeout, the script engine will abort and no further script functions will run. (Integer > 0. Default = 900).
<i>script lines</i>	Script definition with syntax defined by the script engine.

Remarks:

1. At most one script definition may appear in an input file.
2. If the SCRIPT entry occurs in an auxiliary file added with INCLUDE, and the end of that file is reached while reading *script lines*, an implicit ENDSCRIPT will automatically be inserted to end the SCRIPT entry before continuing to read lines from the main input file.
3. If the script encounters an error condition during processing, the entire *GENESIS* run will abort.
4. See [Customization Through Scripts](#) (Sec. 4.16) for an in-depth discussion of the scripting capability available in *GENESIS*.
5. Scripts should be used with caution. Scripts change normal program behavior and may cause unexpected results or errors.

10.2.13 SENSITIVITY

Executive Control Entry: **SENSITIVITY** - Program Flow Control

Description: Stops the program after the first sensitivity calculation.

Format:

SENSITIVITY

Example:

SENSITIVITY

Remarks:

1. The following executive control commands are mutually exclusive (only one may be specified): **ANALYSIS**, **CHECK**, **SENSITIVITY**.
2. Sensitivity results are only available for shape/sizing optimization. This command is not intended to be used in a topology optimization run.
3. This command will be imposed on all sub-models. Any **ANALYSIS**, **CHECK** or **SENSITIVITY** command in a sub-model data file will be ignored.

10.3 Master Data Solution Control

The format of the Solution Control data is free-field. In presenting general formats for each entry embodying all options, the following conventions are used:

1. Upper-case letters must be typed as shown.
2. Lower-case letters indicate that a substitution must be made.
3. Braces { } indicate that a choice of contents is mandatory.
4. Brackets [] contain an option that may be omitted or included by the user.
5. Bold options or values are the default values.
6. Physical data entry consists of information input in columns 1 through 72 of a data entry. Most Solution Control data is limited to a single physical entry.
7. Logical entry may have more than 72 columns with the use of continuation data.
8. Comment lines can be input by using the dollar sign (\$) in the first column of the line.

If the first four characters of a mnemonic are unique relative to all other Solution Control entries, the characters following can be omitted.

10.3.1 APRINT

Solution Control Entry: **APRINT** - Analysis Output Control

Description: Requests analysis results output only at selected design cycles

Format:

$$\text{APRINT} = \left\{ \begin{array}{c} \text{ALL} \\ \text{FIRST} \\ \text{LAST} \\ \text{FLAST} \\ n \\ \text{NONE} \end{array} \right\}$$

Examples:

APRINT = ALL

APRINT = 2

Option	Meaning
ALL	Default. Analysis results output at every Design Cycle.
FIRST	Analysis results output only at the first Design Cycle.
LAST	Analysis results output only at the last Design Cycle.
FLAST	Analysis results output only at the first and last Design Cycle.
n	Analysis results output for the initial, every n-th design cycle after the initial, and the final design cycle (Integer > 0).
NONE	No analysis results are printed.

Remarks:

1. To control printing of DRESP2/3 and TRESP2/3 response values, use the solution control command **DRESP2**.
2. This command will be imposed on all sub-models. Any APRINT command in a sub-model data file will be ignored.

10.3.2 BEGIN BULK

Solution Control Entry: **BEGIN BULK**- Mark the End of Solution Control

Description: Delimits the Solution Control section of the input file from the Bulk Data section.

Format:

BEGIN BULK

Example:

BEGIN BULK

Remarks:

1. The BEGIN BULK delimiter is only required if the master data file has any bulk data entries.

10.3.3 DPRINT

Solution Control Entry: **DPRINT** - Design Output Control

Description: Requests printed design results only at selected design cycles

Format:

$$\text{DPRINT} = \left\{ \begin{array}{c} \text{ALL} \\ \text{FIRST} \\ \text{LAST} \\ \text{FLAST} \\ n \\ \text{NONE} \end{array} \right\}$$

Examples:

DPRINT = ALL

DPRINT = 2

Option	Meaning
ALL	Default. Design results output at every Design Cycle.
FIRST	Design results output only at the first Design Cycle.
LAST	Design results output only at the last Design Cycle.
FLAST	Design results output only at the first and last Design Cycle.
n	Design results output for the initial, every n-th design cycle after the initial, and the final design cycle (Integer > 0).
NONE	No design results are printed.

Remarks:

1. This command will be imposed on all sub-models. Any DPRINT command in a sub-model data file will be ignored.

10.3.4 DRESP2

Solution Control Entry: **DRESP2** - Design Output Request

Description: Requests printing of synthetic response values created with DRESP2, DRESP3, TRESP2 and TRESP3 bulk data statements

Format:

$$\text{DRESP2} = \begin{Bmatrix} \text{YES} \\ \text{NO} \end{Bmatrix}$$

Examples:

DRESP2 = YES

DRESP2 = NO

Option	Meaning
YES	Default. Synthetic responses values will be printed
NO	No synthetic responses values will be printed.

Remarks:

1. This command affects master data response printing only. Sub-model data files should include their own DRESP2 command to control their own synthetic response printing.

10.3.5 ECHO

Solution Control Entry: **ECHO** - Output Control

Description: Requests echo of input data

Format:

$$\text{ECHO} = \left\{ \begin{array}{c} \text{BOTH} \\ \text{SORT} \\ \text{UNSORT} \\ \text{NONE} \end{array} \right\}$$

Alternate Format 1:

$$\text{ECHO} = \left\{ \begin{array}{c} \text{BOTH}(\text{bd1}, \text{bd2}, \dots) \\ \text{SORT}(\text{bd1}, \text{bd2}, \dots) \\ \text{UNSORT}(\text{bd1}, \text{bd2}, \dots) \end{array} \right\}$$

Alternate Format 2:

$$\text{ECHO} = \left\{ \begin{array}{c} \text{BOTH}(\text{EXCEPT } \text{bd1}, \text{bd2}, \dots) \\ \text{SORT}(\text{EXCEPT } \text{bd1}, \text{bd2}, \dots) \\ \text{UNSORT}(\text{EXCEPT } \text{bd1}, \text{bd2}, \dots) \end{array} \right\}$$

Examples:

`ECHO = UNSORT`

`ECHO = SORT(DOPT)`

Option	Meaning
BOTH	Both sorted and unsorted echo will be printed.
SORT	Sorted echo (ordered by type of input data) will be printed. This option prints updated values by initial design data if the optimization data present.
UNSORT	Unsorted echo will be printed. This option prints output that resembles the input data.
NONE	Default. No echo will be printed.
bdi	Bulk data entry names.
EXCEPT	Print all data excluding bulk data entry names listed after EXCEPT

Remarks:

1. Portions of the unsorted echo can be selectively printed using the **ECHOON** and **ECHOOFF** commands. ECHOON starts the printing and ECHOOFF stops the printing. Multiple pairs of ECHOON and ECHOOFF commands are allowed.

10.3.6 ECHOON

Solution Control Entry: **ECHOON** - Output Control

Description: Requests the unsorted echo of input data to be printed in the output file from the ECHOON command until an ECHOOFF command is encountered

Format:

ECHOON

Example:

ECHOON

Remarks:

1. The **ECHOOFF** command stops the printing of unsorted **ECHO**.
2. Multiple pairs of ECHOON and ECHOOFF commands are allowed.

10.3.7 ECHOOFF

Solution Control Entry: **ECHOOFF** - Output Control

Description: Requests that the printing of unsorted echo of input data in the output file be stopped

Format:

ECHOOFF

Example:

ECHOOFF

Remarks:

1. The **ECHOON** command starts the printing of unsorted **ECHO**.
2. Multiple pairs of ECHOON and ECHOOFF commands are allowed.

10.3.8 INCLUDE

Solution Control Entry: **INCLUDE**

Description: Select an external file that contains solution control statements.

Format:

```
INCLUDE 'file name'
```

Alternate Format:

```
INCLUDE = file name
```

Examples:

```
INCLUDE 'SET300.TXT'
```

```
INCLUDE = LC40.INP
```

Option	Meaning
--------	---------

file name	External file name. The user must provide the file name according to the machine installation.
-----------	--

Remarks:

1. The INCLUDE data can be anywhere in the solution control.
2. Multiple INCLUDE data are allowed in the solution control.
3. The external file cannot contain INCLUDE data statements.
4. The external file cannot contain the **BEGIN BULK** delimiter.
5. The file name is limited to 240 characters.
6. If the quoted format is used, and the line does not end with a quote character, additional lines will be read until the closing quote is found. Leading and trailing spaces on continued and continuation lines are discarded.

10.3.9 LINE

Solution Control Entry: **LINE** - Output Control

Description: Defines the number of data lines per printed page

Format:

$$\text{LINE} = \left\{ \begin{matrix} * \\ n \end{matrix} \right\}, \left\{ \begin{matrix} * \\ m \end{matrix} \right\}$$

Examples:

LINE = 35,80

LINE = *,132

Option	Meaning
n	Number of data lines per page (Integer > 0) (default is usually 64)
m	Number of characters per line of output (80 or 132) (default is usually 132)
*	Use default values

Remarks:

1. If no LINE data appears, defaults are used.
2. For 11 inch paper, n=64 is recommended; for 8 1/2 inch paper, n=50 is recommended.
3. This command affects master data printing only. Sub-model data files should include their own LINE command to control their own printing.

10.3.10 OPRINT

Solution Control Entry: **OPRINT** - Optimization Post-processing Output Control

Description: Requests results in optimization post-processing files only at selected design cycles

Format:

$$\text{OPRINT} = \left\{ \begin{array}{c} \text{ALL} \\ \text{FIRST} \\ \text{LAST} \\ \text{FLAST} \\ n \\ \text{NONE} \end{array} \right\}$$

Examples:

OPRINT = ALL

OPRINT = 2

Option	Meaning
ALL	Default. Design results output at every Design Cycle.
FIRST	Design results output only at the first Design Cycle.
LAST	Design results output only at the last Design Cycle.
FLAST	Design results output only at the first and last Design Cycle.
n	Design results output for the initial, every n-th design cycle after the initial, and the final design cycle (Integer > 0).
NONE	No design results are printed.

Remarks:

1. This command will be imposed on all sub-models. Any OPRINT command in a sub-model data file will be ignored.

10.3.11 SUBTITLE

Solution Control Entry: **SUBTITLE** - Output Control

Description: Defines a subtitle which will appear on the third heading line of each page of printer output.

Format:

SUBTITLE = Any Character data

Example:

SUBTITLE = PROBLEM NO. 5-1A

Remarks:

1. If no SUBTITLE entry is supplied, the subtitle line will be blank.
2. The length of the subtitle is limited to 72 characters, including blanks. Another limitation for the length of the Subtitle is that the input data statement should not exceed the 80th column. If it exceeds the 80th column, the excess portion of the SUBTITLE will be ignored and will not be printed in the output file.

10.3.12 TIMES

Solution Control Entry: **TIMES** - Output Control

Description: Requests the printing of CPU and elapsed times.

Format:

$$\text{TIMES} = \left\{ \begin{array}{c} \text{PRINT} \\ \text{SCREEN} \\ \text{BOTH} \\ \text{NONE} \end{array} \right\}$$

Example:

`TIMES = BOTH`

Option	Meaning
PRINT	CPU and elapsed times for each module will be printed to the output file.
SCREEN	CPU and elapsed times for each module will be displayed on the console as <i>GENESIS</i> runs.
BOTH	Times will be printed in the output file and on the screen.
NONE	Default. No times will be printed or displayed.

Remarks:

1. `TIMES=PRINT` or `TIMES=BOTH` will also impose `TIMES=PRINT` on all sub-models.

10.3.13 TITLE

Solution Control Entry: **TITLE** - Output Control

Description: Defines a title which will appear on the second heading line of each page of *GENESIS* printer output.

Format:

TITLE = Any Character data

Example:

TITLE = BODY PANEL DESIGN

Remarks:

1. If no TITLE data is supplied, the second line will be blank.
2. The length of the title is limited to 72 characters, including blanks. Another limitation for the length of the Title is that the input data statement should not exceed the 80th column. If it exceeds the 80th column, the excess portion of the TITLE will be ignored and will not be printed in the output file.

10.3.14 UPRINT

Solution Control Entry: **UPRINT** - Updated Model Output Control

Description: Requests printed an updated model only at selected design cycles

Format:

$$\text{UPRINT} = \left\{ \begin{array}{c} \text{ALL} \\ \text{FIRST} \\ \text{LAST} \\ \text{FLAST} \\ n \\ \text{NONE} \end{array} \right\}$$

Examples:

UPRINT = ALL

UPRINT = 2

Option	Meaning
ALL	Update input file is output at every Design Cycle.
FIRST	Update input file is output only at the first Design Cycle.
LAST	Update input file is output only at the last Design Cycle.
FLAST	Update input file is output only at the first and last Design Cycle.
n	Update input file is output for the initial, every n-th design cycle after the initial, and the final design cycle (Integer > 0).
NONE	Default. No update input file is printed.

Remarks:

1. This command will be imposed on all sub-models. Any UPRINT command in a sub-model data file will be ignored.

10.4 Master Data Bulk Data

The bulk data for design optimization using *GENESIS* is defined in this section. The data is given in alphabetical order.

10.4.1 DCONS

Data Entry: **DCONS** - Design Constraints.

Description: Define design constraints.

Format:

1	2	3	4	5	6	7	8	9	10
DCONS	RID		LB1	UB1					

Example:

1	2	3	4	5	6	7	8	9	10
DCONS	15		-20.0	20.0					

Field Information Description

2	RID	DRESP2 or DRESP3 entry identification number (Integer > 0).
4	LB1	Constraint lower bound imposed on this response quantity. (Real or blank. Default = -1.0E30).
5	UB1	Constraint upper bound imposed on this response quantity. (Real or blank, $LB \leq UB$. default = 1.0E30).

Remarks:

1. A DRESP2 or DRESP3 entry can be referred to by at most one DCONS entry. A DCONS entry cannot reference a DRESP2 or DRESP3 statement that is referred to by a **DCONS2**, **DOBJ** or **DINDEX** statement.
2. Lower bounds < -1.0E+29 and upper bounds > 1.0E+29 are ignored and will not generate constraints. If LBi or UBi are blank, then no constraint will be generated for that bound.

10.4.2 DCONS2

Data Entry: **DCONS2** - Design Constraints.

Description: Define design constraints using scale factors of responses' initial analysis values.

Format:

1	2	3	4	5	6	7	8	9	10
DCONS2	RID		LBF1	UBF1					

Example:

1	2	3	4	5	6	7	8	9	10
DCONS2	15		0.5	1.5.0					

Field Information Description

2	RID	DRESP2 or DRESP3 entry identification number (Integer > 0).
4	LBF1	Scale factor to calculate the constraint lower bound imposed on this response quantity. (Real or blank. Default = no constraint).
5	UBF1	Scale factor to calculate the constraint upper bound imposed on this response quantity. (Real or blank, $LBF \leq UBF$. default = no constraint).

Remarks:

1. A DRESP2 or DRESP3 entry can be referred to by at most one DCONS2 entry. A DCONS2 entry cannot reference a DRESP2 or DRESP3 statement that is referred to by a **DCONS**, **DOBJ** or **DINDEX** statement.
2. If LBFi or UBFi are blank, then no constraint will be generated for that bound.
3. Additional information on DCONS2 can be found in the following chapter:
Relative Constraint Bounds (p. 328)
4. Scaling a negative number with a factor above 1.0 will result in a number less than that with a factor below 1.0. Therefore, care should be taken when using responses that may have negative values to insure that the calculated lower bound is less than the calculated upper bound. If you wish to set a scale factor bound on the magnitude of such a response, then it is recommended to use a DRESP3 with the NORM1 function.

10.4.3 DEQATN

Data Entry: **DEQATN** - Equation Application.

Description: Define equation.

Format:

1	2	3	4	5	6	7	8	9	10
DEQATN	EQID	EQUATION							
+	EQUATION (Cont.)								

Example 1:

1	2	3	4	5	6	7	8	9	10
DEQATN	2	F1(A,B,C,D,R) = A+B*C-(D**3+10.0)+SIN(3.14159*R)							
+	+A**2/(B-C)								

Example 2:

1	2	3	4	5	6	7	8	9	10
DEQATN	3	F(A1, A2, A3, A4, A5, A6, A7, A8, A9, A10, A11, A12, A13, A14, A15, A16, A17, A18,							
+	A19, A20) = A1+A2+A3+A4+A5+A6+A7+A8+A9+A10+A11+A12+A13+A14+A15+A16+A17+								
+	A18+A19+A20								

Example 3: Layered equation.

1	2	3	4	5	6	7	8	9	10
DEQATN	3	F(A1, A2, A3, A4, A5, A6, A7, A8, A9, A10) = A1+A2+A3; G=F+A4+A5+A6 ;							
+	H = F/G+A7+A8+A9+A10								

Field Information Description

2	EQID	Unique equation identification number. (Integer > 0).
3-...	EQUATION	Equation. See remarks.

Remarks:

1. See [Equation Utility](#) (p. 271) for a discussion of the user defined Equation feature.
2. EQUATION consists of the collection of data in fields 3 through 10 on the first entry and fields 2 through 10 on the continuations. The boundaries between these fields are not recognized and the collection is treated as if it were one field.
3. EQUATION may contain embedded blanks.
4. The left-hand side of the first equation must include the complete list of input parameters, enclosed in parentheses, used in the right-hand side of the equations. It must not include the names of the lower level equations.

5. The DEQATN entry is referenced by **DRESP2** data entries. The parameters in the left-hand side of the equation correspond sequentially to NCj and (NRk-LIDRk) on DRESP2 entries.
6. The arithmetic operators in order of precedence are: **, *, /, +, -. The following relational operators may also be used: ==, /=, <, <=, >, >=. The relational operators result in a value of 1.0 if the relation they are testing is true, or a value of 0.0 if the relation is false. The relational operators have lower precedence than the arithmetic operators.
7. The following table lists the available intrinsic functions:

Function	Description
ABS(x)	Absolute value of x
ACOS(x)	Inverse cosine of x (result in radians)
ACOSH(x)	Inverse hyperbolic cosine of x
ASIN(x)	Inverse sine of x (result in radians)
ASINH(x)	Inverse hyperbolic sine of x
ATAN(x)	Inverse tangent of x (result in radians)
ATAN2(y,x)	Inverse tangent of y/x (result in radians, $-\pi$ to π)
ATANH(x)	Inverse hyperbolic tangent of x
ATANH2(y,x)	Inverse hyperbolic tangent of y/x
AVG(x1,x2,...,xn)	Average: $(x1+x2+...+xn)/n$
COS(x)	Cosine of x (x in radians)
COSH(x)	Hyperbolic cosine of x
COTAN(x)	Cotangent of x (x in radians)
DIM(x,y)	Maximum of (0, x-y)
EXP(x)	e raised to power x
INT(x)	Convert x to integer
LOG(x)	Natural (base e) logarithm of x
LOG10(x)	Common (base 10) logarithm of x
LOGX(x,y)	Base x logarithm of y
MAX(x1,x2,...,xn)	Maximum of (x1, x2, ..., xn)
MIN(x1,x2,...,xn)	Minimum of (x1, x2, ..., xn)
MOD(x,y)	Remainder of x/y
PI(x)	π times x
RSS(x1,x2,...,xn)	Square root of sum of squares: $SQRT(x1^{**2} + x2^{**2} + ... + xn^{**2})$

Function	Description
SIGN(x,y)	Absolute value of x times sign of y
SIN(x)	Sine of x (x in radians)
SINH(x)	Hyperbolic sine of x
SQRT(x)	Square root of x
SSQ(x1,x2,...,xn)	Sum of squares: $(x1^{**2} + x2^{**2} + \dots + xn^{**2})$
SUM(x1,x2,...,xn)	Sum: $(x1 + x2 + \dots + xn)$
TAN(x)	Tangent of x (x in radians)
TANH(x)	Hyperbolic tangent of x

8. The relational operators and the functions ABS, DIM, INT, MAX, MIN, MOD, and SIGN should be used with caution, because they can create discontinuities in the function or its derivative. Such discontinuities can cause poor convergence behavior of the optimizer.
9. The maximum number of characters that can be used to define the function names and each of the arguments is 31.
10. Layered equations can be used by separating the equations with semi colons (;). The first equation contains the argument list for all the equations. Equations may reference the value of one or more preceding equations. The value of the last equation is the result of the DEQATN and is the value used by the DRESP2 entry.
11. Constants specified in **DTABLE** data can be used in DEQATN data without passing them through the argument list. If the constant name on DTABLE is the same as an argument list or function name, then the constant in DTABLE is not used in the equation.

10.4.4 DINDEX

Data Entry: **DINDEX** - Design Multi-objective Index Definition.

Description: Define the design objective index function.

Format:

1	2	3	4	5	6	7	8	9	10
DINDEX	RID1		W1	RID2		W2	RID3		W3
+	RID4		W4	-etc.-					

Example:

1	2	3	4	5	6	7	8	9	10
DINDEX	1		0.25						
+	2		0.50						
+	3		0.25						

Field Information Description

2, 5, 8	RIDi	Response entry (DRESP2 or DRESP3) ID. (Integer > 0).
4, 7, 10	Wi	Weighting factor. (Real ≠ 0). See Remark 7.

Remarks:

1. Only one DINDEX statement is allowed in the input data.
2. DRESP2 and DRESP3 entries referenced by the DINDEX data can define only a single response.
3. Only one of **DOBJ** or DINDEX is allowed in the input data.
4. Fields 5-10 may be left blank.

5. If the DOPT parameter, DINDEXM, is 0 or 1, then responses are normalized by their values in the first design cycle. If DINDEXM is 2 or 3, then responses are not normalized.

- DINDEXM = 0 or 1

$$R_i = \frac{\text{Resp}_i}{|\text{Resp}_{0i}|}$$

- DINDEXM = 2 or 3

$$R_i = \text{Resp}_i$$

6. The multi-objective index function is calculated using the following equation:

$$T = \sum_{i=1} f_i$$

If the DOPT parameter DINDEXM is 0 or 2, then the objective function terms are calculated using reciprocals for negative weighting factors. If DINDEXM is 1 or 3, then the compliance index objective function is a straight summation.

- DINDEXM = 0 or 2

$$f_i = \begin{cases} W_i \cdot R_i & \text{if } W_i > 0 \\ \frac{-W_i}{R_i} & \text{if } W_i < 0 \end{cases}$$

- DINDEXM = 1 or 3

$$f_i = W_i R_i$$

Note that in all cases, using a positive weighting factor will drive the corresponding response to be minimized, while a negative weighting factor will drive the corresponding response to be maximized.

7. Typical weighting factor usage;

Response	W_i Sign
FREQ	NEGATIVE (to maximize)
MASS SENERGY	POSITIVE (to minimize)

8. The multi-objective index function is always minimized. To maximize individual responses, use negative weighting factors.

10.4.5 DOBJ

Data Entry: **DOBJ** - Design Objective.

Description: Define a single design objective function.

Format:

1	2	3	4	5	6	7	8	9	10
DOBJ	RID	LABEL		MIN/MAX					

Example:

1	2	3	4	5	6	7	8	9	10
DOBJ	4	VOLUME		MIN					

Field	Information	Description
2	RID	Response entry (DRESP2 or DRESP3) ID. This identifies the response that is to be the objective function. (Integer > 0).
3	LABEL	User defined name for output purposes (Characters or blank).
5	MIN/MAX	Defines minimization (MIN) or maximization (MAX) to be performed. (Default = MIN).

Remarks:

1. The DRESP2 or DRESP3 entry referenced by the DOBJ data can define only a single response.
2. DOBJ cannot be used with any of **DINDEX**.

10.4.6 DOPT

Data Entry: **DOPT** - Optimization parameters.

Description: Define parameters to be used in optimization.

Format:

1	2	3	4	5	6	7	8	9	10
DOPT	DESMAX								
+	NAM1	VAL1	NAM2	VAL2	NAM3	VAL3	NAM4	VAL4	
+	NAM5	VAL5	-etc.-						

Example:

1	2	3	4	5	6	7	8	9	10
DOPT	10								
+	CONV1	0.0001	IPRINT	3	CONVDV	0.01			

Field Information Description

2	DESMAX	Maximum number of approximate optimizations to be performed. (Integer > 0 or blank, Default = 10).
2,4,...	NAMi	Name of additional parameter (see table below for available parameters).
3,5,...	VALi	Value of additional parameter. Real or Integer as indicated by the table below.

Remarks:

1. Only one DOPT statement may appear in the input data file.
1. All DOPT parameter values from the master data file (including default values) will be imposed on all sub-models. Any DOPT command in a sub-model data file will be ignored.
2. See **DOPT** (p. 507) and **DOPT** (p. 677) for parameter name definitions.

10.4.7 DRESP2

Data Entry: **DRESP2** - Create Equation Response Quantities.

Description: Define equation responses that are used in the design, either as an objective function or as constraints.

Format:

1	2	3	4	5	6	7	8	9	10
DRESP2	ID	LABEL	EQID						
+	"DTABLE"	NC1	NC2	NC3	NC4	NC5	NC6	NC7	NC8
+		NC9	-etc.-						
+	"DRESPM"	NR1	MID1	NR2	MID2	NR3	MID3	NR4	MID4
+		NR5	MID5	-etc.-					

Alternate Format 1:

1	2	3	4	5	6	7	8	9	10
DRESP2	ID	LABEL	EQID						
+	"DTABLE"	NC1	NC2	NC3	NC4	NC5	NC6	NC7	NC8
+		NC9	-etc.-						
+	"DRESPML"	NR1	MID1	LID1	NR2	MID2	LID2		
+		NR3	MID3	LID3	-etc.-				

Example:

1	2	3	4	5	6	7	8	9	10
DRESP2	1	EBUCK	3						
+	DTABLE	VAL1	YOUNGS	CVAL4					
+	DRESPM	12	5	2	102	202	302	402	502
+		602	702						

Field Information Description

2	ID	Unique identification number. (Integer > 0). See remark 3.
3	LABEL	User defined label for information only (Characters or blank).
4	EQID	DEQATN entry identification number. (Integer > 0).
2	DTABLE	Word indicating constant identification numbers in DTABLE.
3,4,..	NCj	Name of constant defined in DTABLE input. (Character). See Remark 4.
2	DRESPM	Word indicating Response + Model identification numbers.

3,5,7,9	NRk	DRESP1, DRESP2, DRESP3 or DRESPG identification number (Integer > 0 or Blank) defined in a sub-model. See remark 1.
4,6,8,10	MIDk	Model identification number (Integer > 0).
2	DRESPML	Word indicating Response + Model + Loadcase identification numbers.
3,6	NRk	DRESP1, DRESP2, DRESP3 or DRESPG identification number (Integer > 0 or Blank) defined in a sub-model. See remark 1.
4,7	MIDk	Model identification number (Integer > 0).
5,8	LIDk	Loadcase identification number defined in sub-model. (Integer > 0).

Remarks:

1. A response with ID NRk must be defined by a DRESP1, DRESP2, DRESP3 or DRESPG entry in the sub-model data file associated with model ID MIDk. The sub-model response ID must not be referenced by a DCONS or DCONS2 in the sub-model data file.
2. DRESP2 can refer to response's that have multiple physical responses associated with them. However, each referenced response has to have the same number of values.
3. DRESP2 entries must have unique identification numbers with respect to all other DRESP2 and DRESP3 entries.
4. The sequence in which DTABLE and DRESPM / DRESPML occur is not arbitrary. They must be in the order shown. The DTABLE word along with the NCj list can be omitted if they are not involved in this DRESP2 relationship.
5. If a DRESP2 entry is not referenced by a DOBJ or DCONS/DCONS2 entry, this response is calculated and printed with other DRESP2 values in the DRESP2 response table.
6. The variables identified by NCj and NRk correspond to parameters listed in the left-hand side of the equation on the DEQATN entry identified by EQID. The parameters are assumed to be in the order listed.
7. The name CYCLE can be used in the DTABLE list to pass the current value of the design cycle number (as long as the name CYCLE is not explicitly defined on the DTABLE entry).

10.4.8 DRESP3

Data Entry: **DRESP3** - Create Synthetic Response Quantities.

Description: Define user-subroutine or built-in responses that can be used in the design, either as constraint or as an objective.

Format:

1	2	3	4	5	6	7	8	9	10
DRESP3	ID	LABEL	LIBID or NAME						
+	"DTABLE"	NC1	NC2	NC3	NC4	NC5	NC6	NC7	NC8
+		NC9	-etc.-						
+	"DRESPM"	NR1	MID1	NR2	MID2	NR3	MID3	NR4	MID4
+		NR5	MID5	-etc.-					

Alternate Format 1:

1	2	3	4	5	6	7	8	9	10
DRESP3	ID	LABEL	LIBID or NAME						
+	"DTABLE"	NC1	NC2	NC3	NC4	NC5	NC6	NC7	NC8
+		NC9	-etc.-						
+	"DRESPML"	NR1	MID1	LID1	NR2	MID2	LID2		
+		NR3	MID3	LID3	-etc.-				

Example: User-Subroutine Name - GNSUB3

1	2	3	4	5	6	7	8	9	10
DRESP3	1	EBUCK	3						
+	DTABLE	VAL1	YOUNGS	CVAL4					
+	DRESPM	12	5	2	102	202	302		

Example: Built-in equation AVG3

1	2	3	4	5	6	7	8	9	10
DRESP3	1	SUMM*3	AVG3	4					
+	DRESPM	10	20	30	40	50	60	70	80
+		90	100						

Field Information Description

2	ID	Unique identification number. (Integer > 0). See remark 3.
3	LABEL	User defined label for information only.

4	LIBID or NAME	Library identification number. (Integer > 0). or one of the following built-in function names: SUM1, SUM2, SUM3, SUM4, AVG1, AVG2, AVG3, AVG4, NORM1, NORM2, NORM3, NORM4, MULT1, MULT2, MULT3, MULT4, SAVG1, SAVG2, SAVG3, SAVG4, ASAVG1, ASAVG2, ASAVG3, ASAVG4, PNORM2 and SDEV. See remark 8.
2	DTABLE	Word indicating constant identification numbers in DTABLE.
3,4,..	NCj	Name of constant defined in DTABLE input. (Character). See Remark 4.
2	DRESPM	Word indicating Response + Model identification numbers.
3,5,7,9	NRk	DRESP1, DRESP2, DRESP3 or DRESPG identification number (Integer > 0 or Blank) defined in a sub-model. See remark 1.
4,6,8,10	MIDk	Model identification number (Integer > 0).
2	DRESPML	Word indicating Response + Model + Loadcase identification numbers.
3,6	NRk	DRESP1, DRESP2, DRESP3 or DRESPG identification number (Integer > 0 or Blank) defined in a sub-model. See remark 1.
4,7	MIDk	Model identification number (Integer > 0).
5,8	LIDk	Loadcase identification number defined in sub-model. (Integer > 0).

Remarks:

1. A response with ID NRk must be defined by a DRESP1, DRESP2, DRESP3 or DRESPG entry in the sub-model data file associated with model ID MIDk. The sub-model response ID must not be referenced by a DCONS or DCONS2 in the sub-model data file.
2. DRESP3 can refer to response's that have multiple physical responses associated with them. However, each referenced response has to have the same number of values.
3. DRESP3 entries must have unique identification numbers with respect to all other DRESP2 and DRESP3 entries.
4. The sequence in which DTABLE and DRESPM / DRESPML occur is not arbitrary. They must be in the order shown. The DTABLE word along with the NCj list can be omitted if they are not involved in this DRESP3 relationship.
5. If a DRESP3 entry is not referenced by a DOBJ or DCONS/DCONS2 entry, this response is calculated and printed with other DRESP2/DRESP3 values in the DRESP2 response table.
6. The variables identified by NCj and NRk correspond to values in the VAR array in the GNSUBi user routine. The parameters are assumed to be in the order listed.
7. If a library ID is given, the **DRESP3** executive control command must be used to identify the shared object (DLL) that contains the user calculation subroutines. See **Use of the DRESP3 / TRESP3 capability** (p. 307) for a description of how to create the DRESP3 interface function.

8. The built-in functions are defined as follows (X1, ..., XN are all of the DTABLE, , and DRESPM / DRESPML argument values; N is the number of arguments; LBi = 0.0 and UBi = 1.0:

$$AVG1 = (X1 + X2 + \dots + XN)/N$$

$$AVG2 = (X1^{**2} + X2^{**2} + \dots + XN^{**2})/N$$

$$AVG3 = (X1^{**3} + X2^{**3} + \dots + XN^{**3})/N$$

$$AVG4 = (X1^{**4} + X2^{**4} + \dots + XN^{**4})/N$$

$$MULT1 = X1 * X2 * \dots * XN$$

$$MULT2 = MULT1^{**2}$$

$$MULT3 = MULT1^{**3}$$

$$MULT4 = MULT1^{**4}$$

$$NORM1 = (ABS(X1) + ABS(X2) + \dots + ABS(XN))$$

$$NORM2 = (ABS(X1)^{**2} + ABS(X2)^{**2} + \dots + ABS(XN)^{**2})^{**}(1/2)$$

$$NORM3 = (ABS(X1)^{**3} + ABS(X2)^{**3} + \dots + ABS(XN)^{**3})^{**}(1/3)$$

$$NORM4 = (ABS(X1)^{**4} + ABS(X2)^{**4} + \dots + ABS(XN)^{**4})^{**}(1/4)$$

$$SUM1 = X1 + X2 + \dots + XN$$

$$SUM2 = X1^{**2} + X2^{**2} + \dots + XN^{**2}$$

$$SUM3 = X1^{**3} + X2^{**3} + \dots + XN^{**3}$$

$$SUM4 = X1^{**4} + X2^{**4} + \dots + XN^{**4}$$

$$SAVG1 = ((X1-LB1)/(UB1-LB1) + (X2-LB2)/(UB2-LB2) + \dots + (XN-LBN)/(UBN-LBN))/N$$

$$SAVG2 = (((X1-LB1)/(UB1-LB1))^{**2} + ((X2-LB2)/(UB2-LB2))^{**2} + \dots + (XN-LBN)/(UBN-LBN))^{**2}/N$$

$$SAVG3 = (((X1-LB1)/(UB1-LB1))^{**3} + ((X2-LB2)/(UB2-LB2))^{**3} + \dots + (XN-LBN)/(UBN-LBN))^{**3}/N$$

$$SAVG4 = (((X1-LB1)/(UB1-LB1))^{**4} + ((X2-LB2)/(UB2-LB2))^{**4} + \dots + (XN-LBN)/(UBN-LBN))^{**4}/N$$

$$ASAVG1 = (ABS(X1)/MAX(ABS(UB1),ABS(LB1)) + ABS(X2)/MAX(ABS(UB2),ABS(LB2)) + \dots + ABS(XN)/MAX(ABS(UBN),ABS(LBN)))/N$$

$$ASAVG2 = ((ABS(X1)/MAX(ABS(UB1),ABS(LB1)))^{**2} + (ABS(X2)/MAX(ABS(UB2),ABS(LB2)))^{**2} + \dots + (ABS(XN)/MAX(ABS(UBN),ABS(LBN)))^{**2})/N$$

$$\text{ASAVG3} = ((\text{ABS}(\text{X1})/\text{MAX}(\text{ABS}(\text{UB1}),\text{ABS}(\text{LB1})))^{**3} + \\ (\text{ABS}(\text{X2})/\text{MAX}(\text{ABS}(\text{UB2}),\text{ABS}(\text{LB2})))^{**3} + \dots + \\ (\text{ABS}(\text{XN})/\text{MAX}(\text{ABS}(\text{UBN}),\text{ABS}(\text{LBN})))^{**3})/\text{N}$$

$$\text{ASAVG4} = ((\text{ABS}(\text{X1})/\text{MAX}(\text{ABS}(\text{UB1}),\text{ABS}(\text{LB1})))^{**4} + \\ (\text{ABS}(\text{X2})/\text{MAX}(\text{ABS}(\text{UB2}),\text{ABS}(\text{LB2})))^{**4} + \dots + \\ (\text{ABS}(\text{XN})/\text{MAX}(\text{ABS}(\text{UBN}),\text{ABS}(\text{LBN})))^{**4})/\text{N}$$

$$\text{PNORM2} = \text{SQRT} (\text{MAX}(\text{X1},0.0)^{**2} + \text{MAX}(\text{X2},0.0)^{**2} + \dots + \\ \text{MAX}(\text{XN},0.0)^{**2})$$

$$\text{SDEV} = \text{SQRT} (((\text{X1}-\text{AVG1})^{**2} + (\text{X2}-\text{AVG1})^{**2} + \dots + (\text{XN}-\text{AVG1})^{**2})/\text{N})$$

9. The name CYCLE can be used in the DTABLE list to pass the current value of the design cycle number (as long as the name CYCLE is not explicitly defined on the DTABLE entry).

10.4.9 DSCREEN

Data Entry: **DSCREEN** - Constraint Screening Data.

Description: Define constraint screening data for constraint deletion.

Format:

1	2	3	4	5	6	7	8	9	10
DSCREEN	TYPE	TRS	NSTR						

Example:

1	2	3	4	5	6	7	8	9	10
DSCREEN	STRESS	-0.5	3						

Field	Information	Description
2	TYPE	Type of the constraints for which the screening criteria apply. Must be EQUA.
3	TRS	Truncation threshold. (Real \leq 0.0 or blank, Default = -0.5).
4	NSTR	Maximum number of constraints to be retained per region per load case (per loading frequency). (Integer > 0 or blank) (Default = 20).

Remarks:

- Constraints are retained if

$$\frac{\text{Response} - \text{UBi}}{|\text{UBi}|} \geq \text{TRS}$$

or

$$\frac{\text{LBi} - \text{Response}}{|\text{LBi}|} \geq \text{TRS}$$

where Response is defined on the DRESPi entry and LBi and UBi are defined on the DCONS entry or calculated using DCONS2. If |UBi| is zero, the above equation containing |UBi| is replaced by Response \geq TRS. Similarly, when |LBi| is zero, the equation containing |LBi| is replaced by Response \leq -TRS.

- For **DRESP2** and **DRESP3**, the screening parameters TRS and NSTR for TYPE = EQUA responses are used.
- Only one DSCREEN data statement per TYPE is allowed in the bulk data. Thus, the maximum number of DSCREEN data statements in the bulk data is one.

10.4.10 DTABLE

Data Entry: **DTABLE** - Table Constants.

Description: Define a table of constants that are frequently used in equations.

Format:

1	2	3	4	5	6	7	8	9	10
DTABLE	LABL1	VALU1	LABL2	VALU2	LABL3	VALU3	LABL4	VALU4	
+	LABL5	VALU5	LABL6	VALU6	-etc.-				

Example:

1	2	3	4	5	6	7	8	9	10
DTABLE	PI	3.1416	E	1.0E+6	RHO	0.1	NU	0.3	
+	BK	40.	ALPHA	0.001					

Field Information Description

2,4,..	LABLi	Unique identification label for the constant (non blank characters).
3,5,..	VALUi	Value of the constant (Real).

Remarks:

1. DTABLE data is referenced by **DRESP2** to be used with **DEQATN** or by **DRESP3** to be used in the user-subroutine.
2. Multiple DTABLE entries may appear in the input data.
3. VALUi is referenced by using the corresponding LABLi for CIDI on the DVPROP2 data or NCi on the DRESP2 or DRESP3 data.
4. As an alternative to passing the tabled values through the argument list from DRESP2, the table constant name LABLi can be used directly in DEQATN data in the equation to represent the constant VALUi. In this case, the program will substitute VALUi for LABLi as long as LABLi is distinct from the argument names and function names.

10.4.11 ENDDATA

Data Entry: **ENDDATA** - Mark the End of Bulk Data.

Description: Marks the end of the Bulk Data section of the input file.

Format:

1	2	3	4	5	6	7	8	9	10
ENDDATA									

Example:

1	2	3	4	5	6	7	8	9	10
ENDDATA									

Remarks:

1. Any entries following the ENDDATA entry are ignored.

10.4.12 INCLUDE

Data Entry: **INCLUDE**

Description: Select an external file that contains bulk data statements.

Format:

```
INCLUDE 'file name'
```

Alternate Format:

```
INCLUDE = file name
```

Examples:

```
INCLUDE 'D035.DVS'
```

```
INCLUDE = D035.DVS
```

Option	Meaning
--------	---------

file name	External file name. The user must provide the file name according to the machine installation.
-----------	--

Remarks:

1. The INCLUDE data can be anywhere in the bulk data.
2. Multiple INCLUDE data are allowed in the bulk data.
3. The file name is limited to 240 characters.
4. If the quoted format is used, and the line does not end with a quote character, additional lines will be read until the closing quote is found. Leading and trailing spaces on continued and continuation lines are discarded.

10.4.13 TCONS

Data Entry: **TCONS** - Topology Constraints.

Description: Define topology constraints.

Format:

1	2	3	4	5	6	7	8	9	10
TCONS	RID		LB1	UB1					

Example:

1	2	3	4	5	6	7	8	9	10
TCONS	16	10	0.5	25.0					

Field	Information	Description
-------	-------------	-------------

2	RID	TRESP2 or TRESP3 identification number (Integer > 0).
4	LB1	Constraint lower bound imposed on this response quantity. (Real or blank. Default = -1.0E30).
5	UB1	Constraint upper bound imposed on this response quantity. (Real or blank, $LB \leq UB$. Default = 1.0E30).

Remarks:

1. A TRESP2 or TRESP3 entry can be referred to by at most one TCONS entry. A TCONS entry cannot reference a TRESP2 or TRESP3 that is referred to by a **TCONS2**, **TOBJ** or **TINDEX** statement.
2. Lower bounds < -1.0E+29 and upper bounds > 1.0E+29 are ignored and will not generate constraints. If LBi or UBi are blank, then no constraint will be generated for that bound.

10.4.14 TCONS2

Data Entry: **TCONS2** - Topology Constraints.

Description: Define topology constraints using scale factors of responses' initial analysis values.

Format:

1	2	3	4	5	6	7	8	9	10
TCONS2	RID		LBF1	UBF1					

Example:

1	2	3	4	5	6	7	8	9	10
TCONS2	16	10	0.5	1.0					

Field Information Description

2	RID	TRESP2 or TRESP3 identification number (Integer > 0).
4	LBF1	Constraint lower bound imposed on this response quantity. (Real or blank. Default = no constraint).
5	UBF1	Constraint upper bound imposed on this response quantity. (Real or blank, $LBF \leq UBF$. Default = no constraint).

Remarks:

1. A TRESP2 or TRESP3 entry can be referred to by at most one TCONS2 entry. A TCONS2 entry cannot reference a TRESP2 or TRESP3 that is referred to by a **TCONS**, **TOBJ** or **TINDEX** statement.
2. If LBFi or UBFi are blank, then no constraint will be generated for that bound.
3. Additional information on TCONS2 can be found in the following chapter: **Relative Constraint Bounds** (p. 328)
4. Scaling a negative number with a factor above 1.0 will result in a number less than that with a factor below 1.0. Therefore, care should be taken when using responses that may have negative values to insure that the calculated lower bound is less than the calculated upper bound. If you wish to set a scale factor bound on the magnitude of such a response, then it is recommended to use a TRESP3 with the NORM1 function.

10.4.15 TINDEX

Data Entry: **TINDEX** - Topology Compliance Index Definition.

Description: Define the topology compliance index objective function.

Format:

1	2	3	4	5	6	7	8	9	10
TINDEX	RID1		W1	RID2		W2	RID3		W3
+	RID4		W4	-etc.-					

Example:

1	2	3	4	5	6	7	8	9	10
TINDEX	1		0.33						
+	2		0.33						
+	3		0.33						

Field Information Description

2, 5, 8	RIDi	Response entry (TRESP2 or TRESP3) ID. (Integer > 0).
4,7,10	Wi	Weighting factor. (Real ≠ 0). See Remark 4.

Remarks:

1. Only one TINDEX statement is allowed in the input data.
2. Only one of **TOBJ** or TINDEX is allowed in the input data.

3. If the DOPT parameter, TINDEXM, is 0 or 1, then responses are normalized by their values in the first design cycle. If TINDEXM is 2 or 3, then responses are not normalized.

- TINDEXM = 0 or 1

$$R_i = \frac{\text{Resp}_i}{|\text{Resp}_{0i}|}$$

- TINDEXM = 2 or 3

$$R_i = \text{Resp}_i$$

4. The compliance index objective function is calculated using the following equation:

$$T = \sum_{i=1} f_i$$

If the DOPT parameter TINDEXM is 0 or 2, then the compliance index objective function terms are calculated using reciprocals for negative weighting factors. If TINDEXM is 1 or 3, then the compliance index objective function is a straight summation.

- TINDEXM = 0 or 2

$$f_i = \begin{cases} W_i \cdot R_i & \text{if } W_i > 0 \\ \frac{-W_i}{R_i} & \text{if } W_i < 0 \end{cases}$$

- TINDEXM = 1 or 3

$$f_i = W_i R_i$$

Note that in all cases, using a positive weighting factor will drive the corresponding response to be minimized, while a negative weighting factor will drive the corresponding response to be maximized.

5. Recommended Weighting factor signs are;

Response	W_i Sign
FREQ	NEGATIVE
MASSFR SENERGY	POSITIVE

6. The compliance index objective function is always minimized.

10.4.16 TOBJ

Data Entry: **TOBJ** - Design Objective.

Description: Define topology objective function.

Format:

1	2	3	4	5	6	7	8	9	10
TOBJ	RID	LABEL		MIN/MAX					

Example:

1	2	3	4	5	6	7	8	9	10
TOBJ	14	TOPIND		MIN					

Field	Information	Description
2	RID	Response entry (TRESP2 or TRESP3) ID. This identifies the response that is to be the objective function. (Integer > 0).
3	LABEL	User defined name for output purposes (Characters or blank).
5	MIN/MAX	Defines minimization (MIN) or maximization (MAX) to be performed. (Default = MIN).

Remarks:

1. Only one TOBJ statement is allowed in the input data.
2. TOBJ and **TINDEX** data cannot be used in the same input data.

10.4.17 TRESP2

Data Entry: **TRESP2** - Create Equation Response Quantities.

Description: Define equation responses that are used in the design, either as an objective function or as constraints.

Format:

1	2	3	4	5	6	7	8	9	10
TRESP2	ID	LABEL	EQID						
+	"DTABLE"	NC1	NC2	NC3	NC4	NC5	NC6	NC7	NC8
+		NC9	-etc.-						
+	"TRESPM"	NR1	MID1	NR2	MID2	NR3	MID3	NR4	MID4
+		NR5	MID5	-etc.-					

Alternate Format 1:

1	2	3	4	5	6	7	8	9	10
TRESP2	ID	LABEL	EQID						
+	"DTABLE"	NC1	NC2	NC3	NC4	NC5	NC6	NC7	NC8
+		NC9	-etc.-						
+	"TRESPML"	NR1	MID1	LID1	NR2	MID2	LID2		
+		NR3	MID3	LID3	-etc.-				

Example:

1	2	3	4	5	6	7	8	9	10
TRESP2	1	EBUCK	3						
+	DTABLE	VAL1	YOUNGS	CVAL4					
+	TRESPM	12	5	2	102	202	302	402	502
+		602	702						

Field Information Description

2	ID	Unique identification number. (Integer > 0). See remark 3.
3	LABEL	User defined label for information only (Characters or blank).
4	EQID	DEQATN entry identification number. (Integer > 0).
2	DTABLE	Word indicating constant identification numbers in DTABLE.
3,4,..	NCj	Name of constant defined in DTABLE input. (Character). See Remark 4.
2	TRESPM	Word indicating Response + Model identification numbers.

3,5,7,9	NRk	TRESP1, TRESP2 or TRESP3 identification number (Integer > 0 or Blank) defined in a sub-model. See remark 1.
4,6,8,10	MIDk	Model identification number (Integer > 0).
2	TRESPML	Word indicating Response + Model + Loadcase identification numbers.
3,6	NRk	TRESP1, TRESP2 or TRESP3 identification number (Integer > 0 or Blank) defined in a sub-model. See remark 1.
4,7	MIDk	Model identification number (Integer > 0).
5,8	LIDk	Loadcase identification number defined in sub-model. (Integer > 0).

Remarks:

1. A response with ID NRk must be defined by a TRESP1, TRESP2 or TRESP3 entry in the sub-model data file associated with model ID MIDk. The sub-model response ID must not be referenced by a TCONS or TCONS2 in the sub-model data file.
2. TRESP2 can refer to response's that have multiple physical responses associated with them. However, each referenced response has to have the same number of values.
3. TRESP2 entries must have unique identification numbers with respect to all other TRESP2 and TRESP3 entries.
4. The sequence in which DTABLE and TRESPM / TRESPML occur is not arbitrary. They must be in the order shown. The DTABLE word along with the NCj list can be omitted if they are not involved in this TRESP2 relationship.
5. If a TRESP2 entry is not referenced by a TOBJ or TCONS/TCONS2 entry, this response is calculated and printed with other TRESP2 values in the DRESP2 response table.
6. The variables identified by NCj and NRk correspond to parameters listed in the left-hand side of the equation on the DEQATN entry identified by EQID. The parameters are assumed to be in the order listed.
7. The name CYCLE can be used in the DTABLE list to pass the current value of the design cycle number (as long as the name CYCLE is not explicitly defined on the DTABLE entry).

10.4.18 TRESP3

Data Entry: **TRESP3** - Create Synthetic Response Quantities.

Description: Define user-subroutine or built-in responses that can be used in the design, either as constraint or as an objective.

Format:

1	2	3	4	5	6	7	8	9	10
TRESP3	ID	LABEL	LIBID or NAME						
+	"DTABLE"	NC1	NC2	NC3	NC4	NC5	NC6	NC7	NC8
+		NC9	-etc.-						
+	"TRESPM"	NR1	MID1	NR2	MID2	NR3	MID3	NR4	MID4
+		NR5	MID5	-etc.-					

Alternate Format 1:

1	2	3	4	5	6	7	8	9	10
TRESP3	ID	LABEL	LIBID or NAME						
+	"DTABLE"	NC1	NC2	NC3	NC4	NC5	NC6	NC7	NC8
+		NC9	-etc.-						
+	"TRESPML"	NR1	MID1	LID1	NR2	MID2	LID2		
+		NR3	MID3	LID3	-etc.-				

Example: User-Subroutine Name - GNSUB3

1	2	3	4	5	6	7	8	9	10
TRESP3	1	EBUCK	3						
+	DTABLE	VAL1	YOUNGS	CVAL4					
+	TRESPM	12	5	2	102	202	302		

Example: Built-in equation AVG3

1	2	3	4	5	6	7	8	9	10
TRESP3	1	SUMM*3	AVG3	4					
+	TRESPM	10	20	30	40	50	60	70	80
+		90	100						

Field Information Description

2	ID	Unique identification number. (Integer > 0). See remark 3.
3	LABEL	User defined label for information only.

4	LIBID or NAME	Library identification number. (Integer > 0). or one of the following built-in function names: SUM1, SUM2, SUM3, SUM4, AVG1, AVG2, AVG3, AVG4, NORM1, NORM2, NORM3, NORM4, MULT1, MULT2, MULT3, MULT4, SAVG1, SAVG2, SAVG3, SAVG4, ASAVG1, ASAVG2, ASAVG3, ASAVG4, PNORM2 and SDEV. See remark 8.
2	DTABLE	Word indicating constant identification numbers in DTABLE.
3,4,..	NCj	Name of constant defined in DTABLE input. (Character). See Remark 4.
2	TRESPM	Word indicating Response + Model identification numbers.
3,5,7,9	NRk	TRESP1, TRESP2 or TRESP3 identification number (Integer > 0 or Blank) defined in a sub-model. See remark 1.
4,6,8,10	MIDk	Model identification number (Integer > 0).
2	TRESPML	Word indicating Response + Model + Loadcase identification numbers.
3,6	NRk	TRESP1, TRESP2 or TRESP3 identification number (Integer > 0 or Blank) defined in a sub-model. See remark 1.
4,7	MIDk	Model identification number (Integer > 0).
5,8	LIDk	Loadcase identification number defined in sub-model. (Integer > 0).

Remarks:

1. A response with ID NRk must be defined by a TRESP1, TRESP2 or TRESP3 entry in the sub-model data file associated with model ID MIDk. The sub-model response ID must not be referenced by a TCONS or TCONS2 in the sub-model data file.
2. TRESP3 can refer to response's that have multiple physical responses associated with them. However, each referenced response has to have the same number of values.
3. TRESP3 entries must have unique identification numbers with respect to all other TRESP2 and TRESP3 entries.
4. The sequence in which DTABLE and TRESPM / TRESPML occur is not arbitrary. They must be in the order shown. The DTABLE word along with the NCj list can be omitted if they are not involved in this TRESP3 relationship.
5. If a TRESP3 entry is not referenced by a TOBJ or TCONS/TCONS2 entry, this response is calculated and printed with other TRESP2/TRESP3 values in the DRESP2 response table.
6. The variables identified by NCj and NRk correspond to values in the VAR array in the GNSUBi user routine. The parameters are assumed to be in the order listed.
7. If a library ID is given, the **DRESP3** executive control command must be used to identify the shared object (DLL) that contains the user calculation subroutines. See **Use of the DRESP3 / TRESP3 capability** (p. 307) for a description of how to create the DRESP3 interface function.

8. The built-in functions are defined as follows (X_1, \dots, X_N are all of the DTABLE, and DRESPM / DRESPML argument values; N is the number of arguments; $LB_i = 0.0$ and $UB_i = 1.0$):

$$\begin{aligned} \text{AVG1} &= (X_1 + X_2 + \dots + X_N)/N \\ \text{AVG2} &= (X_1^{**2} + X_2^{**2} + \dots + X_N^{**2})/N \\ \text{AVG3} &= (X_1^{**3} + X_2^{**3} + \dots + X_N^{**3})/N \\ \text{AVG4} &= (X_1^{**4} + X_2^{**4} + \dots + X_N^{**4})/N \end{aligned}$$

$$\begin{aligned} \text{MULT1} &= X_1 * X_2 * \dots * X_N \\ \text{MULT2} &= \text{MULT1}^{**2} \\ \text{MULT3} &= \text{MULT1}^{**3} \\ \text{MULT4} &= \text{MULT1}^{**4} \end{aligned}$$

$$\begin{aligned} \text{NORM1} &= (\text{ABS}(X_1) + \text{ABS}(X_2) + \dots + \text{ABS}(X_N)) \\ \text{NORM2} &= (\text{ABS}(X_1)^{**2} + \text{ABS}(X_2)^{**2} + \dots + \text{ABS}(X_N)^{**2})^{**}(1/2) \\ \text{NORM3} &= (\text{ABS}(X_1)^{**3} + \text{ABS}(X_2)^{**3} + \dots + \text{ABS}(X_N)^{**3})^{**}(1/3) \\ \text{NORM4} &= (\text{ABS}(X_1)^{**4} + \text{ABS}(X_2)^{**4} + \dots + \text{ABS}(X_N)^{**4})^{**}(1/4) \end{aligned}$$

$$\begin{aligned} \text{SUM1} &= X_1 + X_2 + \dots + X_N \\ \text{SUM2} &= X_1^{**2} + X_2^{**2} + \dots + X_N^{**2} \\ \text{SUM3} &= X_1^{**3} + X_2^{**3} + \dots + X_N^{**3} \\ \text{SUM4} &= X_1^{**4} + X_2^{**4} + \dots + X_N^{**4} \end{aligned}$$

$$\begin{aligned} \text{SAVG1} &= ((X_1 - LB_1)/(UB_1 - LB_1) + (X_2 - LB_2)/(UB_2 - LB_2) + \dots + (X_N - LB_N)/(UB_N - LB_N))/N \\ \text{SAVG2} &= (((X_1 - LB_1)/(UB_1 - LB_1))^{**2} + ((X_2 - LB_2)/(UB_2 - LB_2))^{**2} + \dots + (X_N - LB_N)/(UB_N - LB_N))^{**2}/N \\ \text{SAVG3} &= (((X_1 - LB_1)/(UB_1 - LB_1))^{**3} + ((X_2 - LB_2)/(UB_2 - LB_2))^{**3} + \dots + (X_N - LB_N)/(UB_N - LB_N))^{**3}/N \\ \text{SAVG4} &= (((X_1 - LB_1)/(UB_1 - LB_1))^{**4} + ((X_2 - LB_2)/(UB_2 - LB_2))^{**4} + \dots + (X_N - LB_N)/(UB_N - LB_N))^{**4}/N \end{aligned}$$

$$\begin{aligned} \text{ASAVG1} &= (\text{ABS}(X_1)/\text{MAX}(\text{ABS}(UB_1), \text{ABS}(LB_1)) + \text{ABS}(X_2)/\text{MAX}(\text{ABS}(UB_2), \text{ABS}(LB_2)) + \dots + \text{ABS}(X_N)/\text{MAX}(\text{ABS}(UB_N), \text{ABS}(LB_N)))/N \\ \text{ASAVG2} &= ((\text{ABS}(X_1)/\text{MAX}(\text{ABS}(UB_1), \text{ABS}(LB_1)))^{**2} + (\text{ABS}(X_2)/\text{MAX}(\text{ABS}(UB_2), \text{ABS}(LB_2)))^{**2} + \dots + (\text{ABS}(X_N)/\text{MAX}(\text{ABS}(UB_N), \text{ABS}(LB_N)))^{**2})/N \end{aligned}$$

$$\text{ASAVG3} = ((\text{ABS}(\text{X1})/\text{MAX}(\text{ABS}(\text{UB1}),\text{ABS}(\text{LB1})))^{**3} + (\text{ABS}(\text{X2})/\text{MAX}(\text{ABS}(\text{UB2}),\text{ABS}(\text{LB2})))^{**3} + \dots + (\text{ABS}(\text{XN})/\text{MAX}(\text{ABS}(\text{UBN}),\text{ABS}(\text{LBN})))^{**3})/\text{N}$$

$$\text{ASAVG4} = ((\text{ABS}(\text{X1})/\text{MAX}(\text{ABS}(\text{UB1}),\text{ABS}(\text{LB1})))^{**4} + (\text{ABS}(\text{X2})/\text{MAX}(\text{ABS}(\text{UB2}),\text{ABS}(\text{LB2})))^{**4} + \dots + (\text{ABS}(\text{XN})/\text{MAX}(\text{ABS}(\text{UBN}),\text{ABS}(\text{LBN})))^{**4})/\text{N}$$

$$\text{PNORM2} = \text{SQRT} (\text{MAX}(\text{X1},0.0)^{**2} + \text{MAX}(\text{X2},0.0)^{**2} + \dots + \text{MAX}(\text{XN},0.0)^{**2})$$

$$\text{SDEV} = \text{SQRT} (((\text{X1}-\text{AVG1})^{**2} + (\text{X2}-\text{AVG1})^{**2} + \dots + (\text{XN}-\text{AVG1})^{**2})/\text{N})$$

9. The name CYCLE can be used in the DTABLE list to pass the current value of the design cycle number (as long as the name CYCLE is not explicitly defined on the DTABLE entry).

CHAPTER 11

Output Files

- Summary of GENESIS Design Files
- Program Design Output
- Design Cycle History File (pname.HIS)
- Design Information File (pname.OPT)
- Updated Input File (pnameUPDATExx.dat)
- Graph File (pname.html or pname.ps)
- Sensitivity Post-Processing Data (pnamexx.SEN)
- Perturbation Post-Processing Data (pname.DVG)
- Shape Post-Processing Data (pname.SHP)
- Natural Basis (Perturbation) Vector (pname.DVS)
- Grid Basis Vector (pname.DVB)
- AUTORIB Output File (pname.RIB)
- Topology Density File (pname.DNS)
- Topology Density File (pnameDENSxx.ext)
- Topology Isodensity File (pnameTSURFxx.dat)
- Sizing Post-Processing Data (pnameOPOSTxx.ext)
- Shell to Solid Results File (pnameSSOLxx.dat)
- Sensitivities of Guyan Reduced Stiffness Matrix (pnamexxyy.SKA or pnamexxyyyyyyyy.SKA)
- Sensitivities of Guyan Reduced Mass Matrix (pnamexxyy.SMA or pnamexxyyyyyyyy.SMA)

11.1 Summary of *GENESIS* Design Files

INFORMATION	CONTROL	FILE NAME
Design History File	created automatically	<i>pname</i> .HIS
Restart Information File	created automatically	<i>pname</i> .RST
Design Information File	DOPT OPTHS=1	<i>pname</i> .OPT
Updated Input File	UPRINT	<i>pname</i> UPDATExx.dat <i>pname</i> UPDATE=xxxx=.dat
Design History Graph File	GRAPH=YES	<i>pname</i> .html or <i>pname</i> .ps
Sensitivity Post-Processing File	SENSITIVITY=POST	<i>pnamexx</i> .SEN <i>pname</i> =xxxx=.SEN
Natural Basis Vector File	DVSHAPE	<i>pname</i> .DVS
Grid Basis Vector File	DVBASIS	<i>pname</i> .DVB
Perturbation Vectors Post-Processing File	BASIS=POST or PERTURBATION=POST	<i>pname</i> .DVG
Shape Post-Processing File	SHAPE=POST	<i>pname</i> .SHP
AUTORIB Output File	AUTORIB	<i>pname</i> .RIB
Topology Density File (Femb History)	DOPT DNSHS=1	<i>pname</i> .DNS
Topology Density File	DENSITY=POST	<i>pname</i> DENSxx.ext <i>pname</i> DENS=xxxx=.ext
Topology Isodensity File	TSURF	<i>pname</i> TSURFxx.dat <i>pname</i> TSURF=xxxx=.dat
Sizing Post-Processing File	SIZING=POST or THICKNESS=POST	<i>pname</i> OPOSTxx.ext <i>pname</i> OPOST=xxxx=.ext
Shell to Solid Results File	SSOL	<i>pname</i> SSOLxx.dat <i>pname</i> SSOL=xxxx=.dat
Sensitivities of Guyan Reduced Stiffness Matrices	KAASENS=POST	<i>pnamexxyy</i> .SKA <i>pnamexxyyyyyyy</i> .SKA
Sensitivities of Guyan Reduced Mass Matrices	MAASENS=POST	<i>pnamexxyy</i> .SMA <i>pnamexxyyyyyyy</i> .SMA
Element Reliability Results	created whenever probabalistic optimization is performed	<i>pname</i> RELxx.pch

xx = Design Cycle number (if < 100); xxxx = Design Cycle number (if > 99)

yy = Loadcase number (if < 100); yyyyyyy = Loadcase number; ext = op2, op2.gz, pch, unv or PST

11.2 Program Design Output

Design Results

This section contains the design variable values, analysis model property values, shape problem grid point locations, and retained constraint response values for each design cycle. The generation of this data is controlled by the solution control section command **DPRINT**. DPRINT specifies for which design cycles the design results are generated. The default is to produce design results for each design cycle.

The printing of the analysis model properties and grid point locations is controlled by the solution control command **DESIGN**, which can be set to PROPERTY, GRID, BOTH or NONE. The default is NONE.

Sensitivity Results

If sensitivity results are requested with the solution control command **SENSITIVITY**, then the sensitivities of the objective function and the retained constraints with respect to the design variables are printed for each design cycle.

Convergence Information

This section contains the convergence information.

Design Cycle History

This section contains an objective function history table which also includes the maximum constraint violation at each design cycle. Also included is the design variable value history. This is the last section of each run.

11.3 Design Cycle History File (*pname.HIS*)

This file has the name *pname.HIS* and contains the design cycle history. This is comprised of the objective function, maximum constraint violation, and design variable values for each design cycle. This is a formatted readable file and can be used to generate objective function and design variable history plots. This file is also used to restart the optimization. See [Restart from any Previous Design Cycle](#) (p. 322) for more information. The last line of this file contains a completion code and the number of warnings and errors. The value of this code is 2 if there were errors, 1 if there were only warnings, and 0 otherwise. The FORMAT statement used to write this line is (A18,I1,I8,I8).

11.4 Design Information File (*pname.OPT*)

This file contains the shape and sizing design produced by *GENESIS* at each design cycle, including the initial design, and has the name *pname.OPT*. The creation of this file is controlled by the DOPT parameter **OPTTHIS**.

This file contains the design variable values in DVAR input format. It also contains the final design analysis property data (PAXIS, PROD, PBAR, PSHELL, PCOMP, PSHEAR, PCONM3, PELAS, PVECTOR, PSOLID, PVISC, PDAMP, PMASS, PHBDY and PELASH) data for sizing design problems. For shape design problems the file also contains final design grid point locations, in their input coordinate system, in GRID input data format. To get grid point locations in the basic coordinate system, set the DOPT parameter **OPTGRID** to 1. This file can be used to update the input data file with the final design information. If you update the input data file with the .OPT file and shape design variables are present, then you must set the initial values of these design variables to zero if you want to use the updated input data file to rerun the design optimization.

This file can be split into a separate file for each design cycle using the OPTFILE.FOR program that is supplied with *GENESIS*.

If PCOMP data is present, then equivalent PSHELL and MAT2 and/or MAT5 data will follow each PCOMP data in the design information file. The PSHELL, MAT2 and MAT5 data will be preceded by a comment (\$) symbol.

The format mode of PSHELL and/or PBAR data written in the *pname.OPT* file will be dependent on the format mode of the input file, as specified by the executive control command SOL. PBAR and/or PSHELL entries written by *GENESIS* should only be copied to input files with the same SOL COMPATi value as the original input file. Otherwise, the entries must be edited to insure correct interpretation of the data.

11.5 Updated Input File (*pname*UPDATExx.dat)

This file has the name *pname*UPDATExx.dat and contains the analysis input data as updated by the optimization process at design cycle *xx*. This file is in *GENESIS* input file format, so that any preprocessor that can read *GENESIS* input data can load this file. This file is particularly useful for quick visualization of the results of shape or topography optimization. The creation of this file is controlled by the solution control command **UPRINT**. UPRINT specifies for which design cycles the update file is output.

Note that if the design cycle is greater than 99, the filename pattern changes so that the design cycle is 4 digits bracketed by '=' characters: *pname*UPDATE=xxxx=.dat

11.6 Graph File (*pname.html* or *pname.ps*)

This file has the name *pname.html* or *pname.ps* depending on the format selected by the DOPT parameter **GRAPH**. It contains graphs of the objective function history and maximum constraint violation history. The creation of this file is controlled by the solution control command **GRAPH**. A value of YES, the default, causes this file to be created. A value of NO will avoid creation of this file.

11.7 Sensitivity Post-Processing Data (*pnamexx.SEN*)

Data for post-processing the objective function and retained constrained responses can be generated by *GENESIS*. The solution control commands **SENSITIVITY**=POST or **SENSITIVITY**=BOTH are used to generate this data. A separate file is written for each design cycle. The files have the name *pnamexx.SEN* where *xx* is the design cycle number. These are 80 column ASCII formatted files.

Note that if the design cycle is greater than 99, the filename pattern changes so that the design cycle is 4 digits bracketed by '=' characters: *pname=xxxx=.SEN*

The first line of the file is written with the format:

FORMAT (2I8)

and contains the number of retained constraints plus one and the number of independent design variables.

The next lines contain the objective function and retained constrained response data and are written with the format:

FORMAT (I5,1X,A14,I8,I4,I8,2X,I5,2X,A1,2X,E13.6,2X,E13.6)

There is the same number of lines of this data as the first number in first line of the file. There are eight pieces of information in each line of data. The first number is just the number of the line of data. Next is the response type. This can be 'OBJ. FUNCTION', 'DISPLACEMENT', 'VELOCITY', 'ACCELERATION', 'ELEMENT STRESS', 'GRID STRESS', 'ELEMENT STRAIN', 'ELEMENT FORCE', 'MASS', 'VOLUME', 'DRESP2/3', 'FREQUENCY', 'TEMPERATURE', 'STRAIN ENERGY', 'DRESPG', 'EIGENVECTOR', 'INERTIA', 'RFREQ' or 'REVECT'. Next is the load case number. This is zero for the objective function, mass, volume, inertia and DRESPG data. Next is the loading frequency number or the mode number for eigenvector responses. This is zero for responses that are not from dynamic load cases. Next is the grid, element, mode, material, property or DRESP2 ID. This is zero for the objective function, strain energy, and system mass and volume data. Next is the item number. For stress, strain, and force results the item code can be found in the DRESP1 data description in Volume II. For displacements, velocities and accelerations, the item is the component number. For DRESP2/3 data the item number is the number of the response generated by this data. For the objective function the item number is 0. The item number is 1 for mass, volume, strain energy, frequency, and temperature data. Next is the constraint bound indicator. This is "L" for lower bound constraints, "U" for upper bound constraints, and "O" for the objective function. Next is the constraint bound value or 0.0 for the objective function. The last piece of data is the objective function or response value. The data in columns 2-5 is summarized in the table below.

:

11

Column 2	Column 3	Column 5	Column 6
Objective Function	0	0	0
Element Stress, Strain or Force	Loadcase ID	Element #	Item Code
Grid Point Stress	Loadcase ID	Grid #	Item Code
Displacement, Velocity or Acceleration	Loadcase ID	Grid #	Component #
Temperature	Loadcase ID	Grid #	1
Strain Energy	Loadcase ID	0	1
Mass or Volume	0	0 or Material Id or Property ID	1
DRESP2/3	Loadcase ID	DRESP2/3 ID	Response #
Frequency, Reduced Frequency	Loadcase ID	Mode Number	1
Eigenvector, Reduced Eigenvector	Loadcase ID	Grid Number	Component Number
Inertia	0	DRESP1 ID	Item Code
DRESPG	0	DRESPG ID	DRESPG Type Number

:

DRESPG Type Number	Type
1	LENGTH
2	AREA3
3	AREA4
4	VOL4
5	VOL6
6	VOL8
7	ANGLE
8	DGLINE
9	DGPLANE
10	DIFFX
11	DIFFY
12	DIFFZ
13	DISTX
14	DISTY
15	DISTZ

After the objective function and response data is the design variable data. There is the same number of lines of this data as the second number in first line of the file. This data is written with the format:

```
FORMAT(I8,1X,A8,3(1X,E13.7))
```

and contains the design variable ID, label, lower bound, present value, and upper bound.

Finally the sensitivities are written with the format:

```
FORMAT(I8,5(1X,E13.7):/(8X,5(1X,E13.7)))
```

The first number is corresponds to the number of the objective function or response data. Next are the sensitivities of the objective function or response with respect to the independent design variables in the order of the design variable information.

A sample of the output written to this file is shown below.

```

5          5
1  OBJ. FUNCTION      0  0      0  0  O  0.000000E+00  0.233920E+01
2  DISPLACEMENT      2  0      4  1  U  0.100000E+00  0.150787E+00
3  FREQUENCY         1  0      1  1  U  0.650000E+02  0.513941E+02
4  ELEMENT FORCE       7  0      1  12 L -0.160000E+03 -0.156622E+03
5  ELEMENT STRESS     2  0      1  2  U  0.100000E+01  0.249367E+04
11 thick      0.1000000E-01 0.1000000E+00 0.1000000E+01
22 nsm        0.1000000E-01 0.9000000E+00 0.1900000E+01
33 dl         0.1000000E-01 0.1500000E+01 0.1900000E+01
44 d2         0.1000000E-03 0.1000000E-02 0.1900000E+01
5 Shape      -.1000000E+01 0.0000000E+00 0.1000000E+01
1 0.2054950E+02 0.0000000E+00 0.0000000E+00 0.1311726E+04 0.4435050E+00
2 -.3017853E+01 -.1163859E+00 -.7026761E-04 0.0000000E+00 -.1824658E-02
3 0.2885561E+03 0.1983442E+02 0.1197496E-01 -.1440984E+05 -.5535180E+01
4 -.2757058E+04 -.2159335E+02 0.2465222E+00 -.1308386E+05 -.3731953E+02
5 -.4990835E+05 -.1924755E+04 -.1162065E+01 0.0000000E+00 0.2544447E+02

```

11.8 Perturbation Post-Processing Data (*pname.DVG*)

Data for post-processing the basis or perturbation vectors can be generated by *GENESIS*. Either of the two Solution Control parameters, **BASIS** = POST or **PERTURBATION** = POST can be used to generate this data.

The *.DVG file contains the perturbation vectors written as an eigenvector post processing file. The format is selected with the POST executive control command. The design variable ID is used in the file in place of the load case ID. The filename is *pname.DVG* for all formats except POST=PATRAN, for which the filename is *pnamexxxxxxx.DVG* where xxxxxxxx is the design variable ID.

Any post-processing program that can animate eigenvectors can be used to animate the perturbation vectors.

The *.DVG file is written in the basic coordinate system. To get the output in the general coordinate system, use the Solution Control command
POSTOUTPUT = GENERAL.

11.9 Shape Post-Processing Data (*pname.SHP*)

Data for post-processing the shape optimization process can be generated by *GENESIS*. The solution control parameter, **SHAPE** = POST can be used to generate this data.

The *.SHP file contains the shape changes of the model as displacements from the original shape. The SUBCASE number used in the file corresponds to the design cycle times 10 plus 2. The file name is *pname.SHP* for PUNCH format or *pnamexxy.SHP* for PATRAN format. xx is the design cycle number and y is 1 or 2.

When parameters PSMOOTH and MSMOOTH are both ON, then the file contains the changes of shape before and after mesh smoothing. In this case, the SUBCASE number printed corresponds to the design cycle number times 10, plus 1 before mesh smoothing and the design cycle number times 10, plus 2 after mesh smoothing.

Any post-processing program that can animate transient analysis using PUNCH files can be used to animate the shape optimization process with this format.

By default, the *.SHP file is written in the basic coordinate system. To get the output in the general coordinate system, use the Solution Control command POSTOUTPUT = GENERAL.

11.10 Natural Basis (Perturbation) Vector (*pname.DVS*)

The natural basis (perturbation) vectors written in the DVGRID format are printed in the *pname.DVS* file. Also, the corresponding DVAR data is included in the *pname.DVS* file. If the DOPT parameter, BASIS, is set to 1, then the data corresponds to basis vectors. If the DOPT parameter, BASIS, is set to 0, then the data corresponds to perturbation vectors.

To use the information in the *pname.DVS* file, either copy the contents of the file to the input data or use the bulk data statement

```
INCLUDE 'pname.DVS'
```

This file is generated by the **DVSHAPE** solution control command.

11.11 Grid Basis Vector (*pname.DVB*)

The grid basis vectors written in the DVGRID format are printed in the *pname.DVB* file. Also, the corresponding DVAR data is included in the *pname.DVB* file.

To use the information in the *pname.DVB* file, either copy the contents of the file to the input data or use the bulk data statement

INCLUDE '*pname.DVB*'

This file is generated by the **DVBASIS** solution control command.

11.12 AUTORIB Output File (*pname.RIB*)

The **AUTORIB** solution control command enables *GENESIS* to generate candidate rib stiffener elements over an entire shell surface. The generated GRID, element connectivity, element property and material property data are printed in the AUTORIB output file named *pname.RIB*.

To use the information in the *pname.RIB* file, either copy the contents of the file to the input data or use the bulk data statement:

```
INCLUDE 'pname.RIB'
```

A subsequent topology optimization of the candidate stiffener elements can reveal the best place to add rib or bead stiffening to the shell surface.

11.13 Topology Density File (*pname.DNS*)

11

This file contains the topology design variable values (volume fraction) for each element at each design cycle and has the name *pname.DNS*. The creation of this file is controlled by the DOPT parameter **DNSHIS**.

The format of this file is the ETA/FEMB "History" format. This file can be read directly by the FEMB pre/post-processor to animate the topology results. Also, this file is used by the VR&D supplied dns2dat program which creates a *GENESIS* data file with elements associated with different properties according to the value of the topology design variables. This makes it easy to visualize the topology results in any pre/post-processor that allows the user to turn properties on and off.

The file begins with the header:

```
FORMAT ( '$ HISTORY' )
FORMAT ( '$ VOL. FRAC.' )
```

Each design cycle begins with the header:

```
FORMAT ( '$ TIME           ',I3) design cycle
FORMAT ( 'STEP',I6) design cycle + 1
FORMAT ( 'BEGIN BULK' )
FORMAT ( 'ENDDATA' )
```

Within each cycle, a block is written for each element type. The header for each element type block is as follows:

```
FORMAT ( '$TITLE    = MATERIAL DISTRIBUTION' )
FORMAT ( '$SUBTITLE= TOPOLOGY DESIGN' )
FORMAT ( '$ELEMENT STRESSES' )
FORMAT ( '$REAL OUTPUT' )
FORMAT ( '$SUBCASE ID=',6X,I6) design cycle + 1
FORMAT ( '$ELEMENT TYPE=',4X,I8) element type code
```

The element type codes are the same as for PUNCH format post-processing files and are listed on the Analysis Reference manual. The results are written one element per line with the following format:

```
FORMAT ( I8,E12.4) element ID, topology design variable value
```

11.14 Topology Density File (*pnameDENSxx.ext*)

To facilitate visualizing topology results on various third-party post-processors, the topology design variable values (volume fraction) for each element may optionally be printed in a post processing file named *pnameDENSxx.ext*, where *pname* is set to the base of the input filename, *xx* is the design cycle number, and *ext* is set according to the file format. The file format is set by the POST executive control command. The extension of the filename is based on the format: *op2* for OUTPUT2 format, *op2.gz* for compressed OUTPUT2 format, *pch* for PUNCH format, *unv* for IDEAS format, *PST* for BINARY, FORMAT, PLOT or PATRAN format.

Note that if the design cycle is greater than 99, the filename pattern changes so that the design cycle is 4 digits bracketed by '=' characters: *pnameDENS=xxxx=.ext*

The solution control command, **DENSITY** = POST, is used to generate this output.

The design variable values are printed using the element strain energy analysis result style.

11.15 Topology Isodensity File (*pname*TSURFxx.dat)

The isodensity file is named *pname*TSURFxx.dat and contains smoothed surface meshes obtained from the topology densities.

The format of this file is the same as *GENESIS* input format. This file can be read directly by any preprocessor that can read *GENESIS* input data. This file can be created during a *GENESIS* topology optimization run or by a TSURFACE run. In the optimization run this file is created using the information of the last design cycle. In the TSURFACE run the isodensity file is written for a user specified design cycle. The density levels can be selected by the user using the solution control command **TSURF**.

Note that if the design cycle is greater than 99, the filename pattern changes so that the design cycle is 4 digits bracketed by '=' characters: *pname*TSURF=xxxx=.dat

11.16 Sizing Post-Processing Data (*pnameOPOSTxx.ext*)

To facilitate visualizing sizing / topometry optimization results, real-valued element property fields may optionally be printed in a post processing file named *pnameOPOSTxx.ext*, where *pname* is set to the base of the input filename, *xx* is the design cycle number, and *ext* is set according to the file format. The file format is set by the POST executive control command. The extension of the filename is based on the format: *op2* for OUTPUT2 format, *op2.gz* for compressed OUTPUT2 format, *pch* for PUNCH format, *unv* for IDEAS format, *PST* for BINARY, FORMAT, PLOT or PATRAN format.

Note that if the design cycle is greater than 99, the filename pattern changes so that the design cycle is 4 digits bracketed by '=' characters: *pnameOPOST=xxxx=.ext*

The solution control command, **SIZING** = POST, is used to generate this output. Currently, only the OUTPUT2 and PUNCH formats are supported for this option.

The solution control command, **THICKNESS** = POST, can also be used to generate the OPOST file. When this command is used, only thickness values for CQUAD4 / CTRIA3 elements are output. In this case, all POST formats are supported.

The element property values are printed using the element strain energy analysis result style. The output will consist of records for each element property field containing data for all elements of the corresponding type. By default, output records will only be generated for a given property field if at least one property has design data associated to that property field. If the optimization parameter **OPOST** is set to 1, then all element property fields are output even if they are never designed.

In the output records, the loadcase field is used as a code to indicate which element property field the record contains, as indicated in the following table. FID values are the same as those used on the **DVPROP1** entry. ELTYPE is the bar element type code as used on the **DVPROP3** entry:

LOADCASE CODE	ELEMENT PROPERTY FIELD
1	PSHELL thickness / PCOMP total thickness
10*(ELTYPE+2) + i	DVPROP3 (PBARL) cross section dimension i
1000 + FID	PROD field
2000 + FID	PBAR field
3000 + FID	PSHELL field
5000 + FID	PELAS field
6000 + FID	PCONM3 field
7000 + FID	PHBDY field
8000 + FID	PSHEAR field

LOADCASE CODE	ELEMENT PROPERTY FIELD
9000 + FID	PDAMP field
10000 + FID	PMASS field
11000 + FID	PVISC field
12000 + FID	PELASH field
13000 + FID	PCOMP field
14000 + FID	PAXIS field
15000 + FID	PVECTOR field
17000 + FID	PBUSH field
18000 + FID	PK2UU field
19000 + FID	PM2UU field

11.17 Shell to Solid Results File (*pnameSSOLxx.dat*)

To facilitate visualizing shell sizing optimization results, shell elements can be converted to solid elements that reveal the thickness using the **SSOL** solution control command. The SSOL file is named *pnameSSOLxx.dat* and contains grid data and solid element connectivity (CHEXA and CPENTA) that reveal the thicknesses of CQUAD4 and/or CTRIA3 elements of the input data file.

Note that if the design cycle is greater than 99, the filename pattern changes so that the design cycle is 4 digits bracketed by '=' characters: *pnameSSOL=xxxx=.dat*

The format of this file is the same as *GENESIS* input format. This file can be read directly by any preprocessor that can read *GENESIS* input data.

11.18 Sensitivities of Guyan Reduced Stiffness Matrix (*pnamexxyy.SKA* or *pnamexxyyyyyyyy.SKA*)

If **MPRINT** = ALL, the sensitivities of the Guyan reduced stiffness matrices are printed for every design cycle and for every Guyan reduction loadcase that has the **KAASENS** = POST solution control command using the following procedure:

```

FOR Every design cycle:
  LOOP over all guyan reduction loadcases with KAASENS = POST
    OPEN the PNAMEXXYY.SKA file
    LOOP over the independent design variables
      WRITE(LUN9) ICYCLE,LOADID,NEQR,NDVI
      WRITE(LUN9) (IWORK(1,J),IWORK(2,J),J=1,NEQR)
      WRITE OUT REDUCED MATRIX BY COLUMNS OF LOWER TRIANGLE
      ILAST = 0
      DO 10 J = 1,NEQR
        NROW = NEQR - J + 1
        IFIRST = ILAST + 1
        ILAST = IFIRST + NROW - 1
        WRITE(LUN9) (SKMAA(I),I=IFIRST,ILAST)
      10  CONTINUE
    CONTINUE
  CLOSE the PNAMEXXYY.SKA file
  CONTINUE

```

Where,

PNAME is the project name as define in the ID executive control command.

XX indicates the cycle number.

YY indicates the loadcase number(when the loadcase id is lower than 100).

YYYYYYYY indicates the loadcase number (when the loadcase id larger than 99).

LUN9 is the unit number use to write the unformatted sequential access file

PNAMEXXYY.SKA or PNAMEXXYYYYYYYY.SKA is the unformatted sequential access file that contains the sensitivities of Guyan reduced stiffness matrix

ICYCLE is the design cycle number

LOADID is the load case number

NEQR is the number of ASET degrees of freedoms

NDVI is the number of independent design variables

IWORK(1,J) contains the grid number (J=1,NEQR)

IWORK(2,J) contains the component number (J=1,NEQR)

SKMAA is a double precision array that contains the guyan reduced stiffness matrix

If MPRINT=FIRST, the same procedure is used except that only the sensitivities of the reduced stiffness matrices for design cycle zero are printed.

11.19 Sensitivities of Guyan Reduced Mass Matrix (*pnamexxyy.SMA* or *pnamexxyyyyyyyyyy.SMA*)

If **MPRINT** = ALL, the sensitivities of the Guyan reduced mass matrices are printed for every design cycle and for every Guyan reduction loadcase that has the **MAASENS** = POST solution control command using the following procedure:

```

FOR Every design cycle:
  LOOP over all guyan reduction loadcases with MAASENS = POST
    OPEN the PNAMEXXYY.SMA file
    LOOP over the independent design variables
      WRITE(LUN9) ICYCLE,LOADID,NEQR,NDVI
      WRITE(LUN9) (IWORK(1,J),IWORK(2,J),J=1,NEQR)
      WRITE OUT REDUCED MATRIX BY COLUMNS OF LOWER TRIANGLE
      ILAST = 0
      DO 10 J = 1,NEQR
        NROW = NEQR - J + 1
        IFIRST = ILAST + 1
        ILAST = IFIRST + NROW - 1
        WRITE(LUN9) (SKMAA(I),I=IFIRST,ILAST)
      10  CONTINUE
    CONTINUE
    CLOSE the PNAMEXXYY.SMA file
  CONTINUE

```

Where,

PNAME is the project name as define in the ID executive control command.

XX indicates the design cycle number.

YY indicates the loadcase number(when the loadcases id is lower than 100).

YYYYYYYY indicates the loadcase number (when the loadcase id larger than 99).

LUN9 - is the unit number use to write the unformatted sequential access file.

PNAMEXXYY.SMA or PNAMEXXYYYYYYYY.SMA is the unformatted sequential access file that contains the sensitivities of Guyan reduced mass matrix.

ICYCLE is the design cycle.

LOADID is the load case number.

NEQR is the number of ASET degrees of freedoms.

NDVI is the number of independent design variables.

IWORK(1,J) contains the grid number (J=1,NEQR).

IWORK(2,J) contains the component number (J=1,NEQR).

SKMAA is a double precision array that contains the guyan reduced stiffness matrix.

If MPRINT=FIRST, the same procedure is used except that only the sensitivities of the reduced mass matrices for design cycle zero are printed.

11.20 Element Reliability Results File (*pnameRELxx.pch*)

11

Whenever reliability optimization is performed, reliability results associated to element results will be printed in a file named *pnameRELxx.pch*. Where, *pname* is the *GENESIS* project name and *xx* is to the corresponding design cycle number. This is a post-processing file using the PUNCH element strain energy format, where the 3 values per element are assigned alternative meanings. The first value is the maximum probability of failure of any retained constraint associated to that element in any loadcase. The second value is the maximum probability of failure index of any retained constraint associated to that element in any loadcase. The third value is the loadcase ID of the loadcase where the maximum probability of failure index of any retained constraint associated to that element occurs or 0 if none. Note that the probability of failure index is calculated as the probability of failure divided by the allowable probability of failure. A value greater than 1.0 for the probability of failure index means that the probability of failure is greater than allowed.

Note that if the design cycle is greater than 99, the filename pattern changes so that the design cycle is 4 digits bracketed by '=' characters: *pnameREL=xxxx=.pch*

CHAPTER 12

References

- o [References](#)

12.1 References

1. Schmit, L. A., "Structural Design by Systematic Synthesis," Proc. 2nd Conference on Electronic Computation, American Society of Civil Engineers, New York, 1960, pp. 105-132.
2. Schmit, L. A. and Farshi, B., "Some Approximations Concepts for Structural Synthesis," AIAA Journal, Vol. 12, May 1974, pp. 692-699.
3. Schmit, L. A. and Miura, H. "Approximation Concepts for Efficient Structural Synthesis," NASA CR-2552, 1976.
4. Vanderplaats, G. N. and Salajegheh, E., "A New Approximation Method for Stress Constraints in Structural Synthesis," AIAA Journal, Vol. 27, No. 3, March 1989, pp. 352-358.
5. Canfield, R. A., "High-Quality Approximation of Eigenvalues in Structural Optimization," AIAA Journal, Vol. 28, No. 6, June 1990, pp. 1116-1122.
6. NASTRAN User's Manual, NASA SP-222(08), Volume 1, National Aeronautics & Space Administration, June 1986.
7. DOT User's Manual, VR&D, 1767 South Eight Street, Suite 100, Colorado Springs, CO 80906.
8. Vanderplaats, G. N., Numerical Optimization Techniques for Engineering Design; with Applications, 3rd Ed., Vanderplaats Research & Development, 1999.
9. Leiva, J.P. and Watson, B.C., "Automatic Generation of Basis Vectors for Shape Optimization in the GENESIS program," Proceedings of the 7th AIAA/USAF/NASA/ISSMO Symposium on Multidisciplinary Analysis and Optimization. St. Louis, MO, Sep. 2-4, 1998, pp 1115-1122.
10. Leiva, J.P., Watson, B.C., and Kosaka, I., "Modern Structural Optimization Concepts Applied to Topology Optimization," Proceedings of the 40th AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics, and Material Conference. St. Louis, MO, April-12-15, 1999, pp 1589-1596.
11. Kosaka, I., Charpentier, C., and Watson, B., 'An Interface Between SDRC I-DEAS and the GENESIS Structural Analysis and Optimization Code, Proceedings of the 8th AIAA/USAF/NASA/ISSMO Symposium at Multidisciplinary Analysis and Optimization, Long Beach, CA September 6-8, 2000.
12. Leiva, J.P., and Watson, B.C., 'Buckling Finite Element Analysis and Optimization in GENESIS, Proceedings of the 8th AIAA/USAF/NASA/ISSMO Symposium at Multidisciplinary Analysis and Optimization, Long Beach, CA September 6-8, 2000.
13. Vanderplaats, G., 'Very Large Scale Optimization', Proceedings of the 8th AIAA/USAF/NASA/ISSMO Symposium at Multidisciplinary Analysis and Optimization, Long Beach, CA September 6-8, 2000

References

14. Leiva, J.P. , Industrial Applications using Structural Optimization with GENESIS, Proceedings of the 4th World Congress of Structural and Multidisciplinary Optimization, Dalian, China, June 4-8, 2001
15. Rastogi, N., Ghosh, D. K., and Vanderplaats, G., Discrete Optimization Capabilities in GENESIS Structural Analysis and Optimization Software, Proceedings of the 9th AIAA/ISSMO Symposium on Multidisciplinary Analysis and Optimization, Atlanta, GA, September 4-6, 2002.
16. Leiva, J. P., Watson, B., Kosaka, I., and Vanderplaats, G., Dynamic Finite Element Analysis and Optimization in GENESIS, Proceedings of the 9th AIAA/ISSMO Symposium on Multidisciplinary Analysis and Optimization, Atlanta, GA, September 4-6, 2002.
17. Leiva, J. P., Ghosh, D. K., Rastogi, N., A New Approach in Stacking Sequence Optimization of Composite Laminates using GENESIS Structural Analysis and Optimization Software, Proceedings of the 9th AIAA/ISSMO Symposium on Multidisciplinary Analysis and Optimization, Atlanta, GA, September 4-6, 2002.
18. Leiva, J. P., Methods for Generating Perturbation Vectors for Topography Optimization of Structures, Proceedings of the 5th World Congress of Structural and Multidisciplinary Optimization, Venice, Italy, May 19-23, 2003.
19. Leiva, J. P., Topometry Optimization: A New Capability to Perform Element by Element Sizing Optimization of Structures, presented at the 10th AIAA/ISSMO Symposium on Multidisciplinary Analysis and Optimization, Albany, NY, August 30 - September 1, 2004.
20. Leiva, J. P., Freeform Optimization: A New Capability to Perform Grid by Grid Shape Optimization of Structures, presented at the 6th China-Japan-Korea Joint Symposium on Optimization of Structural and Mechanical Systems, Kyoto, Japan, June 22-25, 2010.

APPENDIX **A**

Diagnostic Information

- **Diagnostic Information: Design**
- **The DIAG Command**

1.1 Diagnostic Information: Design

This chapter describes the diagnostic information available by including the DIAG command in the Executive Control section of the data.

1.2 The DIAG Command

Diagnostic information is available through the use of the DIAG command in the executive control section of *GENESIS*. The DIAG command has the format:

```
DIAG = d1,d2,d3,d4,...
```

where the d_i are the diagnostic switches. This command is used to tell *GENESIS* to write the contents of internal scalars, tables, and matrices into the output data file. This main reason this option is available is for program debugging. However, certain items may be of value to the user. An abbreviated list of the result of each diagnostic switch is presented here. In most cases each diagnostic switch also turns on all other diagnostic switches with the same smaller ones digit and the same tens and hundreds digit. For example 167 also turns on 161 through 166.

Following is a list of diagnostic numbers and the information that they produce.

Overall

- 11 CPU time spent in each module.
- 12 CPU and elapsed wall clock time spent in each module.

Data Manager

- 21 Print the data manager dictionary when over write is detected.
- 23 Check the void tables.
- 25 Initialize all the data blocks that are created without initialization to a special value; 99999.
- 28 Check data block unreleased by enforcing every released data block to be stored out of core only.
- 29 Check read only unreleased by comparing the data block with its out of core version when releasing.

Input/Output

- 32 Print information about the status of files open with unknown status.
- 33 Print information about every file's opening and closing.
- 36 Print the elemental stiffness and mass matrices.

Design Progress

The following information is directed to the error unit number (usually unit 0. unit 7 for HP) and can be written to the terminal, console, LOG file, or output file.

- 81 Print design cycle number, objective function value and maximum constraint violation for each design cycle, as well as the convergence flag and completion code.
- 82 Print message after the execution of each program module that includes CPU time and at the start of each design cycle.
- 83 Print design cycle history.
- 84 Print wall clock time spent in each module.
- 85 Print the problem summary.
- 87 Print all of the above plus more detailed summaries.

Input Data

- 121 Print messages before and after renumbering.
- 125 Solution control, analysis, and design data blocks before renumbering.
- 126 Solution control, analysis, and design data blocks after renumbering.

Topography Data

- 131 Print basic summaries.
- 132 Print constants.
- 138 Print data blocks.

Design Data Preprocessing

- 162 Design problem scalars.
- 165 Design tables.
- 166 Record pointers and shape design tables.
- 167 Response tables.
- 168 Flag tables and equation information.

Constraint Screening

- 411 Approximation comparison table for each design cycle.
Move limit reduction message. Master retained
constraint table.
- 412 Constraint screening scalars.
- 413 Small fixed length constraint screening tables.
- 414 Constraint screening tables of length of # of dresp's
- 415 Constraint screening tables of length of # of retained
constraints.
- 416 Retained response tables.
- 417 Constraint screening tables of length of # of
responses.
- 418 Constraint screening tables of length of # of
constraints.

Sensitivity Analysis

- 501 Print header indicating the beginning and ending of
the solution process.
- 502 Print the currently processing design variable number
to indicate the progress of the sensitivity analysis.
- 503 Print the sensitivity matrix.
- 504 Print the newly created data blocks.
- 505 Print the displacement derivative vectors in terms of
equation numbers.
- 507 Print all the intermediate response tables

Approximate Problem

- 601 Print approximate problem optimum constraint table.
- 602 Approximate problem scalars.
- 603 Approximate problem small fixed length tables and tables length of # of dv's.
- 604 Approximate problem tables of length of # of prop's and # of grid move limits.
- 605 Approximate problem tables of length of # of retained constraints.
- 606 Approximate problem tables of length of # of coordinates.

Analysis Model Update

- 611 Print GRID locations in the Basic, rather than the CP, coordinate system in the ".OPT" file.
This option has been superseded by the Parameter OPTGRID.
- 616 New grid and property tables for each design cycle.

Soft Convergence

- 631 Design variable change information for each design cycle.

Mesh Smoothing

- 901 Use the old (version 10.1 or earlier) mesh smoothing algorithm. Print information about element distortion before and after mesh smoothing.
- 902 Use the old (version 10.1 or earlier) mesh smoothing algorithm. Correct connectivity of 2D planar meshes so that all elements have normals in the same direction.
- 903 Use the old (version 10.1 or earlier) mesh smoothing algorithm. Smooth mesh even if less than 3% of the elements are distorted.
- 904 Same as 902 plus 903.

- 961 Print statistics about the mesh quality measures during mesh smoothing.

Finish Up

- 992 Print maximum disk space used by SCRATCH files. Scratch files are the files used by GENESIS that are deleted after a successful run.

▼ Index

GENESIS

Design Manual

A

- Analysis Capabilities 6
- ANALYSIS Command 427, 766
- Approximation Concepts 19
- APRINT Command 445, 452
- Automatic Generation of Basis Vectors 181
- Automatic Generation of Grid Stress Constraints 147
- Automatic Generation of Perturbation Vectors 157
- Automatic Generation of Stress Constraints 143
- Automatic Generation of Surface Stress Constraints 149
- AUTORIB Command 445, 453
- AUTORIB Output File (.RIB) 841
- Auxiliary Model 183, 188

B

- Background 3
- Bar Element Library 623
 - Type 1 - Square 625
 - Type 10 - Tee 649
 - Type 11 - Angle 652
 - Type 2 - Rectangle 627
 - Type 3 - Circle 630
 - Type 4 - Tube 632
 - Type 5 - Spar 635
 - Type 6 - Box3 (Constant Thickness Box) 637
 - Type 7 - Box4 (Two Thickness Box) 640
 - Type 8 - I Beam 643
 - Type 9 - Rail 645
- BASIS Command 446, 455
- Basis Designs 73
- Basis Post-Processing Data (.DVG) 837
- Beam Element Library 107
 - Type 1 - Square 109
 - Type 10 - Tee 119
 - Type 11 - Angle 120
 - Type 2 - Rectangle 110
 - Type 3 - Circle 111
 - Type 4 - Tube 112
 - Type 5 - Spar 113

- Type 6 - Box3 (Constant Thickness Box) 114
- Type 7 - Box4 (Two Thickness Box) 115
- Type 8 - I Beam 116
- Type 9 - Rail 117
- Bulk Data
 - Shape and Sizing Design 29
 - Topology 33
- Bulk Data - Design
 - Automatic Generation of Basis Vectors 181
 - Automatic Generation of Perturbation Vectors 157
 - Automatic Stress Constraint Generation 143, 147, 149
 - Bar Element Library 623
 - Basic Sizing Design Capabilities 39
 - Constraint Screening (DSCREEN) 132
 - Constraints 256
 - Constraints (DCONS) 68
 - Constraints (TCONS) 377
 - Design Element Library (DVPROP3) 105
 - Design Variable Definition (DVAR) 41
 - Design Variable Linking (DLINK) 47
 - DOMAIN and DVGRIDC Data Statement 159, 181
 - DRESP1 252
 - DVBASIS Data 187
 - DVGRIDC Data Statement 162, 180
 - DVSHAPE Data 182, 186
 - Element Stress Constraints 144
 - External Analysis 299
 - General Nonlinear Responses (DRESP2) 128
 - General Nonlinear Responses (TRESP2) 397
 - Geometric Responses (DRESPG) 59
 - Grid Locations using Basis Vectors 73
 - Grid Locations using Grid Perturbations 86
 - Input for User Defined Design Elements (DLIB) 286
 - Library of Beam and Plate Elements (DVPROP3) 107
 - Linear Variable/Property Relations (DVPROP1) 42
 - Move Limits 135
 - Nonlinear Variable/Property Rel. (DVPROP2) 103
 - Objective Function 256
 - Objective Function (DMATCH) 69
 - Objective Function (DOBJ) 63, 64
 - Objective Function (TINDEX) 372
 - Objective Function (TOBJ) 372
 - Optimization Process Control (DOPT) 139
 - Plate Element Library 623
 - Relationships Among Data 483
 - Relationships Among Data for Topology Optimization 671
 - Structural Responses

- Topometry Split 251
- Structural Responses (DRESP1) 49
- Structural Responses (TRESP1) 365
- Bulk Data - Shape and Sizing Design
 - Constraints 31
 - Design Variable to Grid Relations 29
 - Design Variable to Property Relations 29
 - Design Variables 29
 - Objective Function 30
 - Parameters 31
 - Relationships Among Design Data 32
 - Responses 30
- Bulk Data - Topology Design
 - TCONS 68, 377
 - TINDEX 64, 372
 - TOBJ 372
 - TPROP 352
 - TRESP1 367
 - TSYM1 378
 - TSYM2 378
 - TSYM3 379
- Bulk Data Statements - Analysis
 - GRID 138, 153
- Bulk Data Statements - Design
 - DCONS 68, 143, 147, 149, 155, 485
 - DCONS2 487
 - DEQATN 103, 155, 272, 273, 275, 489, 674
 - DLIB 286, 287, 494
 - DLINK 47, 496
 - DMATCH 69, 497, 499
 - DMATCH2 69
 - DOBJ 63, 501
 - DOMAIN 160, 502
 - DOPT 507, 677
 - DRESP1 49, 522
 - DRESP2 128, 155, 554
 - DRESP3 307, 558
 - DRESPG 59, 564
 - DRESPU 299, 569
 - DSCREEN 132, 570, 687
 - DSPLIT 582
 - DTABLE 588, 692
 - DTGRID 589
 - DVAR 41, 45, 135, 263, 265, 266, 596
 - DVGRID 73, 86, 87, 90, 91, 93, 598
 - DVGRIDC 160, 162, 179, 180, 181, 600
 - DVPROP1 42, 45, 137, 603

- DVPROP2 103, 137, 612
- DVPROP3 105, 137, 621
- DVPROP4 125, 663
- DVSET 665
- DVSET1 666
- GRID 138, 153
- TCONS 693
- TCONS2 695
- TINDEX 699
- TOBJ 702
- TPROP 703
- TPROPC 706
- TRESP2 397, 720
- TRESP3 309, 724
- TSYM1 732
- TSYM2 741
- TSYM3 750
- TVAR 396, 759
- Bulk Data Statements - Shape and Sizing Design
 - DCONS 31
 - DEQATN 29, 30
 - DINDEX 30
 - DLIB 29
 - DLINK 29
 - DMATCH 30, 31, 35
 - DMATCH2 30
 - DOBJ 30
 - DOMAIN 29
 - DOPT 31
 - DRESP1 30
 - DRESP2 30
 - DRESP3 30
 - DRESPG 30
 - DRESPU 30
 - DSCREEN 31
 - DSHAPE 29
 - DSPLIT 29
 - DTABLE 29, 30, 34
 - DTGRID 29
 - DVAR 29, 41
 - DVGRID 29
 - DVGRIDC 29
 - DVPROP1 29
 - DVPROP2 29
 - DVPROP3 29, 46
 - DVPROP4 29, 125
 - DVSET 29

- DVSET1 29
- Bulk Data Statements - Topology Design
 - DEQATN 34
 - DOPT 35
 - DSCREEN 34
 - DTABLE 34
 - TCONS 34
 - TINDEX 34
 - TOBJ 34
 - TPROP 34
 - TRESP1 34
 - TRESP2 34
 - TRESP3 34
 - TSYM1 35
 - TSYM2 35
 - TSYM3 35
 - TVAR 35

C

- Compliance Index 374
- Composite Elements 70
- Composite Failure Index 70
- Composite Property Design 125
- Constraint Screening 16, 132
- Constraints 31, 68
 - Automatic Generation of 143, 147, 149
- Convergence 22
- Coordinate Systems
 - Shape Optimization 151

D

- DCONS2 328, 329
- DENS File 843
- DENSITY Command 448, 456
- Design Bulk Data 484
- Design Capabilities 7
- DESIGN Command 446, 457
- Design Element Library
 - Adding To 285
 - DLIBPROP 288
 - DLIBSTRESS 292
- Design Variable Linking 47
- Design Variable to Property Relations 42
- Design Variable to Property Relationship 105
- Design Variable to Property Relationships 103

- Design Variables 29, 41
- Diagnostic Information 857
- Discrete Optimization 27, 261
- DLIB Executive Control Command 428, 432, 434
- DPRINT Command 445, 458
- DRESP2 Command 445, 459
- DRESP3 Executive Control Command 430
- DVBASIS Command 447, 460
- DVGRID Command 461
- DVSHAPE Command 447, 462

E

- Equation Utility 271
 - Error Messages 281
 - Use with DRESP2 Data 278
 - Use with DVPROP2 Data 276
- Executive Control Statements
 - ANALYSIS 427
 - DIAG 858
 - DLIB 428, 432, 434
 - DRESP3 430
 - GNUSER 431
 - RESTART 433
 - SCRIPT 435
 - SENSITIVITY 436
 - SSOLID 437
 - TOPOLOGY 438
 - TSURFACE 439
- External Analysis Programs 296

F

- Fabrication Constraints 358
 - Types 380
- Fabrication Constraints 237, 242, 247, 378
- Freeform Optimization 192

G

- General Features 4
- General Responses 128, 397
- Geometric Responses 59
- GNUSER 297
- Gradient Calculations 17
- GRAPH Command 447, 464
- Graph File 832

- Grid Basis Vector Output (.DVB) 840
- Grid Basis Vectors 444
- Grid BasisVectors (DVBASIS Data) 187
- Grid Perturbation Approach 86
- Guyan Reduced Stiffness Matrix Output (.SKA) 848

H

- Hard Convergence 22
- History (.HIS) File 829

K

- KAASENS Command 465

L

- Laminated Composites 70

M

- MAASENS Command 466
- Matching Eigenvector Components 327
- Matching Measured Data 325
- Mesh Smoothing 330
- Mode Tracking 323
- MODTRK Command 467
- Move Limits 21, 135
 - Shape Optimization 152
 - Topology Optimization 357
 - Topometry Optimization 250
- MPRINT Command 445, 468

N

- Natural Basis (Perturbation) Vectors 182
- Natural Basis Vector Output (.DVS) 839
- Natural Basis Vectors 443

O

- Objective Function 30, 63, 64, 69
- OPOST File 845
- OPRINT Command 445, 469
- OPT File 830
- Optimization

- Approximation Concepts 19
- Constraint Screening 16
- Constraints 12
- Convergence 22
- Discrete Design Variables 261
- Discrete Variables 27
- Equation Utility 271
- Freeform 192
- General Concepts 11
- Gradient Calculations 17
- Hard Convergence 22
- Move Limits 21
- Objective Function 12
- Soft Convergence 22
- Stress Ratio Method 24
- Topology 8, 33, 347
- Topometry 222
- Optimization Process 15
- Original Model 185, 190
- Output Files 827
 - AUTORIB Output File (.RIB) 841
 - Design Cycle History (.HIS) 829
 - Design Information File (.OPT) 830
 - Graph File (.ps) 832
 - Grid Basis Vector (.DVB) 840
 - Guyan Reduced Stiffness Matrix (.SKA) 848
 - Natural Basis (Perturbation) Vector (.DVS) 839
 - Perturbation Vectors 837
 - Program Output 828
 - Sensitivity Files (.SEN) 833
 - Shape Post-Processing Data (.SHP) 838, 845
 - Shell Thickness File (OPOST) 845
 - Shell-to-Solid File (SSOL) 847
 - Topology Density File (.DNS) 842
 - Topology Density File (DENS) 843
 - Topology Isodensity File (TSURF) 844
 - Updated Input File 831, 850

P

- Parameters, Analysis
 - MODTRK 323
 - MSMOOTH 330
 - PSMOOTH 330
- Parameters, Optimization
 - ADJOIN 17, 142, 508
 - BASIS 508

.... BDMEM 520, 685
.... CONV1 512, 681
.... CONV2 512, 681
.... CONVCN 512, 681
.... CONVDV 512, 681
.... CONVLC 512
.... CONVPR 512
.... DABOBJ 681
.... DDAMIN 516
.... DDCMIN 516
.... DDELA 516
.... DDELC 516
.... DDELL 516
.... DDELP 516
.... DDLMIN 516
.... DDPMIN 516
.... DELOBJ 681
.... DELP 510
.... DELT 684
.... DELX 510, 684
.... DINDEXM 508
.... DNSHIS 685
.... DPMIN 510
.... DR1MV 508
.... DSCDOT 514
.... DSTART 515
.... DTMIN 684
.... DVGTOL 513
.... DVINIT 520, 685
.... DVINIT2 515
.... DVTOL 513
.... DXFRAC 510
.... DXMIN 510, 684
.... FDCHMU 513
.... FDCHU 513
.... FILTER 679
.... GMAX 512, 681
.... GRAPH 514
.... ICOMPAPP 508
.... IMATCH 518
.... IPEN 515
.... IPRINT 514, 684
.... ISDMAX 517
.... ISHLAPP 509
.... ISRMAX 517
.... ISRMET 517
.... ITMAX 519, 685

- ITRMOP 519, 685
- IUGRAD 509
- LAMASNS 509
- METHOD 509, 678
- MODAPP 509
- NDSCRT 515
- OPOST 514
- OPTGRID 514
- OPTHIS 514
- OPTM 509, 678
- PENLTD 515
- PMULTD 515
- PTOL 513
- RCOMPAPP 510
- RMATCH 518
- RSHLAPP 510
- SGENEL 513
- SK2UU 513
- SM2UU 513
- STRDOT 517
- SYMMET 518
- SYMTOL 518
- TINDEXM 678
- TMIN 684
- PERTURBATION Command 446, 470
- Perturbation Design Approach 86
- Perturbation Post-Processing Data (.DVG) 837
- Plate Element Library 107, 623
- Type 12 - Solid (through the thickness) 122, 657
- Type 13 - Sand (Sandwich Plate) 123, 659
- Type 14 - Sand2 (Two Thickness Sandwich Plate) 124, 661

R

- References 853
- Relative Constraint Bounds 328
- Responses 30
- Restart Capability 321
- RESTART Command 433

S

- SCRIPT Executive Control Command 435
- Scripting 333
- SENSITIVITY Command 445
- SENSITIVITY Executive Control Command 436
- Sensitivity Output File (.SEN) 833

- SENSITIVITY Solution Control Command 472
- Shape Bulk Data 484
- SHAPE Command 446, 473
- Shape Optimization 72
 - Automatic Generation of Perturbation Vectors 157
 - Basis Designs 73
 - Effect of Nonrectangular Coordinate Systems 156
 - Grid Basis Vectors 187
 - Move Limits 152
 - Natural Basis Vectors 182
 - Unexpected Results 84
- Shape Optimization Bulk Data 29, 32
- Shape Post-Processing Data (.SHP) 838, 845
- Shell Thickness File (OPOST) 845
- Shell-to-Solid Output (SSOL) 847
- Sizing Bulk Data 484
- SIZING Command 446, 474
- Sizing Optimization 39
- Sizing Optimization Bulk Data 29, 32
- Soft Convergence 22
- Solution Control
 - Design Output Control Commands 446
 - Design Output Requests 449, 450
 - Grid Basis Vectors 444
 - Guyan Reduction Matrices Output Requests 449
 - Natural Basis (Perturbation) Vector Output Requests 449
 - Natural Basis (Perturbation) Vectors 443
 - Other General Output Control Commands 445
 - Output Control 449
 - Output Selection 449
 - Topology Output Control Command 448
 - Topology Output Requests 450
- Solution Control Data
 - APRINT 452
 - AUTORIB 453
 - BASIS 455
 - DENSITY 456
 - DESIGN 457
 - DPRINT 458
 - DRESP2 459
 - DVBASIS 460
 - DVGRID 461
 - DVSHAPE 462
 - GRAPH 464
 - KAASENS 465
 - MAASENS 466
 - MODTRK 467

- MPRINT 468
- OPRINT 469
- PERTURBATION 470
- SENSITIVITY 472
- SHAPE 473
- SIZING 474
- SSOL 475
- THICKNESS 476
- TSURF 477
- TVAR 478
- UPRINT 479
- Solvers, Linear Equation 6
- SSOL Command 475
- SSOL File 847
- SSOLID Command 437
- STRDOT Optimizer 26
- Stress Constraints
 - Automatic Generation of 143, 147, 149
- Stress Ratio 24, 331
- Structural Responses 49, 128, 397
- SUMMARY Command 445

T

- TCONS2 328, 329
- THICKNESS Command 476
- Topolmetry Fabrication Constraints 242
- Topology
 - Fabrication Constraints
 - Cyclic Symmetry 386
 - Extrusion 388
 - Filling 389, 391
 - Mirror Symmetry 385
- Topology Bulk Data 671
- Topology Capabilities 8
- TOPOLOGY Command 438
- Topology Density Output (.DNS) 842
- Topology Density Output (DENS) 843
- Topology Fabrication Constraints 378
- Topology Isodensity Output (TSURF) 844
- Topology Optimization 33, 347
- Topometry Fabrication Constraints 237, 247
- Topometry Optimization 222
- Topometry Responses
 - DRESP1 252
 - DRESP2 253
 - DRESP3 254, 255

- TSURF Command 448, 477
- TSURF File 844
- TSURFACE Command 439
- TVAR Command 478

U

- UPDATE File 831, 850
- UPRINT Command 479

