Superelement User's Guide
Contents

Proprietary & Restricted Rights Notice .......................................................... 7

Introduction and Fundamentals ........................................................................ 1-1
Why Use Superelements? .............................................................. 1-2
  Reduced Cost ........................................................................ 1-3
  Quicker Turnaround ....................................................... 1-3
  Reduced Risk ..................................................................... 1-3
  Large Problem Capabilities ........................................... 1-3
  Partitioned Input and Output .................................. 1-3
  Security .............................................................................. 1-4
Fundamentals of Superelement Analysis ............................................. 1-4
Partitioned Solutions ....................................................................... 1-6
Example to Show the Use of Superelements in Static Analysis ............ 1-8
  Conventional Analysis ............................................. 1-9
  Superelement Analysis ........................................ 1-11
  Superelement 1 ................................................................ 1-12
  Superelement 2 ............................................................. 1-15
  Residual Structure ..................................................... 1-16

How to Define a Superelement ...................................................................... 2-1
Defining a Superelement Using Partitioned Bulk Data ....................... 2-2
  Defining PARTs ................................................................ 2-2
  The Bulk Data Section Using PARTs .......................... 2-3
  Format of the Input File When PARTs Are Used .......... 2-3
  Automatically Connecting PARTs to Other Components of the Model 2-12
  Constraints on Connecting Points .................................. 2-12
  Manually Defining Exterior Points for a PART ............... 2-17
  Moving and/or Rotating PARTs ..................................... 2-17
Defining Superelements in the Main Bulk Data Section ....................... 2-22
  Superelement Definition ............................................... 2-22
  Interior Versus Exterior Points ..................................... 2-23
  Bulk Data Partitioning .................................................. 2-23
  Example of Bulk Data Partitioning ............................... 2-24
  The Superelement MAP - SEMAP .................................. 2-28
Defining External Superelements ......................................................... 2-30
  External Superelement Procedure Overview ................. 2-30
  NX Nastran Input File Requirements for Component Solutions .... 2-32
  NX Nastran Input File Requirements for System Solutions .... 2-35
  External Superelement Loads ....................................... 2-36
  External Superelement Examples .................................. 2-37

Single-Level Superelement Analysis ......................................................... 3-1
Contents

Introduction ......................................................... 3-2
Simple Input for Single-Level Analysis .................................. 3-2
  Single-Level Analysis Using Main Bulk Data Superelements ............... 3-3
  Example of Single-Level Analysis .................................. 3-3
Quick Review ......................................................... 3-6
Single-Level Analysis When PARTs Are Present .......................... 3-7

Loads, Constraints, and Case Control in Static Analysis .................. 4-1
Introduction .................................................................. 4-2
Loads in Static Analysis ............................................... 4-2
  Thermal Loadings in Superelement Analysis .......................... 4-3
Boundary Conditions .................................................. 4-4
  Single-Point Constraints in Superelement Analysis .................. 4-4
  MPCs and R-Type Elements in the Main Bulk Data Section ............... 4-5
Case Control in Superelement Analysis .................................. 4-7
  The SUPER Command-Case Control Partitioning ............... 4-8
  Conventional Case Control ........................................ 4-9
  Condensed Case Control ........................................... 4-9
  Superelement Case Control ........................................ 4-10
  One Loading Condition—Expanded Case Control .................... 4-10
  Parameters in the Case Control Section .............................. 4-18

Inertia Relief Analysis Using Superelements ............................ 5-1
Introduction .................................................................. 5-2
  The Concept of Inertia Relief ....................................... 5-2
  Interface for Inertia Relief Using Superelements ................... 5-2
  Example of Inertia Relief Using Superelements ..................... 5-3
  Selected Output from inrel2.f06 .................................. 5-9

Multiple Loadings in Static Analysis .................................... 6-1
Introduction .................................................................. 6-2
  How Case Control is Internally Partitioned and Used .............. 6-2
  Examples for Multiple Loading Conditions ........................... 6-2

Multilevel Superelement Analysis ....................................... 7-1
The Concept of Multilevel Analysis .................................... 7-2
Comparison of Single- and Multilevel Analysis .......................... 7-2
  Single-Level Analysis .............................................. 7-2
  Multilevel Analysis ................................................ 7-6
User Interface for Multilevel Superelements ............................. 7-18
  Multilevel Processing When the Model is Main Bulk Data Only ........ 7-18
  Automatic Creation of the Processing Tree .......................... 7-18
  Manual Definition of the Processing Tree for a Model Defined in the Main Bulk Data Section Only .......................... 7-19
  Multilevel Processing When the Model Uses PARTs ............... 7-20
Example: Multilevel Problem Solved by Hand ......................... 7-20
  SUPERELEMENT 2 .................................................. 7-23
  Residual Structure .................................................. 7-24

4  Superelement User's Guide
### Contents

Examples of Multilevel Superelements ................................. 7-28
  Input File multi1.dat-Multilevel Superelements Using Main Bulk Data Only 7-30
  Input File seset.mult ................................. 7-31
  Input File multi2.dat-Multilevel Superelements Using PARTs 7-32

Output in Superelement Static Analysis ................................. 8-1
  Sorted Bulk Data for PARTs ................................. 8-2
  Boundary Grid Search Output for PARTs .............................. 8-3
  Superelement Definition Table ................................. 8-4

Introduction to Superelements in Dynamic Analysis ............................. 9-1
  Description of Dynamic Reduction Procedures .............................. 9-2
  Reduction Methods Used for Superelements .............................. 9-3
    Static Condensation (Guyan Reduction) .............................. 9-3
    Dynamic Reduction ................................. 9-4
    Fixed-Boundary Dynamic Reduction .............................. 9-7
    Performing Data Recovery for Superelement 2 ............................. 9-18
    Repeating the Same Process for Superelement 1 ............................. 9-19
    Free-Free Dynamic Reduction .............................. 9-19
    Mixed-Boundary Dynamic Reduction .............................. 9-20
    CMS with Exterior DOF in the C- and/or R-SET ............................. 9-20

Input and Output for Dynamic Reduction ................................. 10-1
  Case Control for Dynamic Reduction .............................. 10-2
  Case Control for Dynamic Reduction .............................. 10-2
  Single-Level Dynamic Reduction ................................. 10-4
    Bulk Data Entries for Single-Level Dynamic Reduction of Main Bulk Data Superelements 10-4
    Samples of Single-Level Dynamic Reduction for Main Bulk Data Superelements 10-6
      File cantbeam.dat - Input Model for the Examples 10-6
    File seg10_b1.dat - Fixed Boundary CMS of the Superelements 10-8
    File seg10-c1 - Free-Free CMS of the Superelements 10-9
    File seg10_d1.dat - Mixed Boundary CMS 10-11
    Bulk Data Entries for Single-Level Dynamic Reduction of PARTs 10-13
    Samples for Single-Level Dynamic Reduction Using PARTs 10-14
      File cantp1.dat - Bulk Data for PART 1 10-15
      File cantp2.dat - Bulk Data for PART 2 10-15
      File seg10p_a.dat - Static Reduction Using PARTs 10-16
      File seg10p_b1.dat - Fixed Boundary CMS Using PARTs 10-17
      File seg10p_c1.dat - Free-Free CMS Using PARTs 10-20
      File seg10p_d1.dat - Mixed-Boundary CMS Using PARTs 10-23
  Multilevel Dynamic Reduction ................................. 10-24
    Bulk Data Entries for Multilevel Dynamic Reduction 10-24
    Multilevel Dynamic Reduction for a Model With No PARTs 10-25
    Bulk Data Entries for Multilevel Dynamic Reduction When PART Superelements Exist 10-32
    Sample of Multilevel CMS Using PARTs 10-34
Dynamic Loading on Superelements ........................................... 11-1

Case Control Used to Define Dynamic Loadings on Superelements using LOADSET - LSEQ ........................................ 11-2
  LOADSET = i ....................................................................... 11-2
  Bulk Data ............................................................................. 11-2
Example Demonstrating Dynamic Loads on Superelements ........................................ 11-3
  Dynamic Loadings Applied on PARTs .................................... 11-7

References .............................................................................. 12-1
Proprietary & Restricted Rights Notice

© 2014 Siemens Product Lifecycle Management Software Inc. All Rights Reserved.

This software and related documentation are proprietary to Siemens Product Lifecycle Management Software Inc. Siemens and the Siemens logo are registered trademarks of Siemens AG. NX is a trademark or registered trademark of Siemens Product Lifecycle Management Software Inc. or its subsidiaries in the United States and in other countries.

NASTRAN is a registered trademark of the National Aeronautics and Space Administration. NX Nastran is an enhanced proprietary version developed and maintained by Siemens Product Lifecycle Management Software Inc.

MSC is a registered trademark of MSC.Software Corporation. MSC.Nastran and MSC.Patran are trademarks of MSC.Software Corporation.

All other trademarks are the property of their respective owners.

TAUCS Copyright and License

TAUCS Version 2.0, November 29, 2001. Copyright (c) 2001, 2002, 2003 by Sivan Toledo, Tel-Aviv University, stoledo@tau.ac.il. All Rights Reserved.

TAUCS License:

Your use or distribution of TAUCS or any derivative code implies that you agree to this License.

THIS MATERIAL IS PROVIDED AS IS, WITH ABSOLUTELY NO WARRANTY EXPRESSED OR IMPLIED. ANY USE IS AT YOUR OWN RISK.

Permission is hereby granted to use or copy this program, provided that the Copyright, this License, and the Availability of the original version is retained on all copies. User documentation of any code that uses this code or any derivative code must cite the Copyright, this License, the Availability note, and "Used by permission." If this code or any derivative code is accessible from within MATLAB, then typing "help taucs" must cite the Copyright, and "type taucs" must also cite this License and the Availability note. Permission to modify the code and to distribute modified code is granted, provided the Copyright, this License, and the Availability note are retained, and a notice that the code was modified is included. This software is provided to you free of charge.

Availability (TAUCS)

As of version 2.1, we distribute the code in 4 formats: zip and tarred-gzipped (tgz), with or without binaries for external libraries. The bundled external libraries should allow you to build the test programs on Linux, Windows, and MacOS X without installing additional software. We recommend that you download the full distributions, and then perhaps replace the bundled libraries by higher performance ones (e.g., with a BLAS library that is specifically optimized for your machine). If you want to conserve bandwidth and you want to install the required libraries yourself, download the lean distributions. The zip and tgz files are identical, except that on Linux, Unix, and MacOS, unpacking the tgz file ensures that the configure script is marked as executable (unpack with tar zxvpf), otherwise you will have to change its permissions manually.
Chapter 1: Introduction and Fundamentals

- Why Use Superelements?
- Fundamentals of Superelement Analysis
- Partitioned Solutions
- Example to Show the Use of Superelements in Static Analysis
In finite element analysis, demand for computer resources will always exceed existing capabilities. In the early days of computers, when engineers were solving $3 \times 3$ problems by hand, computers were able to handle problems as large as $11 \times 11$. Once engineers discovered this ability, the size of engineering problems quickly grew to exceed the capacity of the existing systems. This process has repeated itself time and time again. Today modern computers are capable of solving problems involving more than 1,000,000 equations with 1,000,000 unknowns, which is still not enough to satisfy the needs of many engineers.

This limit on hardware resources, combined with budget restrictions (large runs can be time-consuming and expensive), limits the ability of engineers to solve large, complicated problems. A solution to these problems, in terms of both hardware and budget, is the use of superelements in NX Nastran.

Using superelements, you can not only analyze large models (including those that exceed the capacity of your hardware), but can become more efficient in performing the analysis, thus allowing more design cycles or iterations in the analysis.

The principle used in superelement analysis is often referred to as substructuring. That is, the model is divided into a series of components (superelements), each of which is processed independently, resulting in a set of reduced matrices that describe the behavior of the superelement as seen by the rest of the structure. These reduced matrices for the individual superelements are combined to form an assembly (or residual) solution. The results of the assembly solution are then used to perform data recovery (calculation of displacements, stresses, etc.) for the superelements.

In static analysis the theory used in superelement processing is exact. In dynamics the reduction of the stiffness is exact, but approximations occur during the reduction of the mass and damping matrices. These approximations may be improved by using a method called component modal synthesis, which is described in "Introduction to Superelements in Dynamic Analysis".

This user’s guide is intended to be tutorial in format: Emphasis is on how to use superelements, not on the theory of superelements. Sufficient theory is presented for those who wish to understand the operations. Hand-solved samples are included to help you understand the operations involved when superelements are used. Sample NX Nastran input files and selected output are also presented at appropriate points for clarity.

This guide is arranged so that an experienced finite element analyst can start at the beginning and read only the information applicable to the type of analysis desired. Overall information on superelements is presented first, followed by information for static analysis, followed by dynamics and other features. You should read this guide from the beginning, because much of the information presented in the section on statics is applicable in subsequent sections (similar to engineering itself); however, an engineer should be able to read the applicable sections without having to read unnecessary information.

1.1 Why Use Superelements?

Efficiency is the primary reason to use superelements. A finite element model is rarely analyzed only once. Often the model is modified and re-analyzed time and time again. Without using superelements, each analysis can cost the price of a complete solution. Doing this can deplete a budget in a short time. What follows is a partial list of the advantages of superelements.
Reduced Cost

Instead of solving the entire model each time, superelements offer the advantage of incremental processing. On restarts, this advantage is magnified by the need to process only the parts of the structure directly affected by the change. This means that if you think ahead when defining superelements, you can achieve performance improvements on the order of anywhere from 2 to 30 times faster than non-superelement methods. The use of split databases allows control of disk usage and reduces the computer resource requirements for individual runs without sacrificing the accuracy of the results obtained.

Quicker Turnaround

Because superelements can be processed individually with less computer resources required than a complete, non-superelement solution, it is often possible to submit individual superelement processing runs using fast queues (or even on different computers), rather than waiting and running the complete problem at once using an overnight queue.

Reduced Risk

Processing a model without using superelements is an all-or-nothing proposition. If an error occurs, the entire model must be processed again once the error is corrected. With superelements, each superelement need be processed only once, unless a change requires reprocessing the superelement. If an error occurs during processing, only the affected superelement and the residual structure (final superelement to be processed) need be reprocessed. The superelements that did not have an error do not need to be processed again until a change is made to those superelements.

Large Problem Capabilities

All computers have hardware limits. NX Nastran is designed so that problem size will not be limited by the program. This means that limits on available disk space or memory are the only limitations that you should encounter. When the size of a model becomes too large to be processed on a computer without using superelements, you can use split databases for incremental processing and copy onto tape database information that is not needed until data recovery. This process frees file space and reduces disk usage and storage costs. For example, a user with a model containing over 200,000 degrees of freedom (DOFs) was able to solve the problem on a computer that had limited disk space (the largest problem that could be processed without superelements was 15,000 DOFs) by dividing it into superelements.

Partitioned Input and Output

Because superelements can be processed individually, separate analysis groups can model individual parts of the structure and perform checks and assembly analysis without information from other groups. An excellent example is Space Station Freedom, which has many contractors working on the structure. Each contractor models his own components and sends either complete or reduced models to a system integrator, who assembles the models to represent the many possible configurations, performs analysis for each configuration, and sends results back to the individual contractors for their use. The partitioned output format used in superelements allows for segmented data recovery, that is, data recovery can be requested for only the desired segments of the structure. Also, with the partitioned output format, you can have selective data recovery for each group in the case of many groups working on a structure.
Security

Many companies work on proprietary or secure projects. These projects can range from keeping a new design from the competition to working on a highly confidential defense program. Even when working on secure programs, there is a need to send a representation of the model to others so that they can perform a coupled analysis of an assembly that incorporates the component. The use of external superelements allows users to send reduced boundary matrices that contain no geometric information about the actual component-only mass, stiffness, damping and loads as seen at the boundary. Upon receiving a set of reduced matrices in any format that can be read by NX Nastran, an engineer can define an external superelement using those matrices and attach the foreign structure to his model.

1.2 Fundamentals of Superelement Analysis

Superelement analysis can be described as a form of substructuring. That is, a model can be divided into superelements by the user in such a way that NX Nastran processes each superelement independently of all other superelements.

The processing of each superelement results in a reduced set of matrices (mass, damping, stiffness, and loading) that represent the properties of the superelement as seen at its connections to adjacent structures. Once all superelements have been processed, these reduced matrices are assembled in what is known as the residual structure, and the assembly solution is performed. Data recovery for each superelement is performed by expanding the solution at the attachment points, using the same transformation that was used to perform the original reduction on the superelement.

Superelements may consist of physical data (elements and grid points) or may be defined as an image of another superelement or as an external superelement (a set of matrices from an external source to be attached to the model).

The following figures illustrate the possible types of superelement. Figure 1-1 shows a model of a portion of a gear. The physical model of one tooth can be represented as a superelement. This type is called a primary superelement—one where the actual geometry for the superelement is defined in the bulk data.

Other gear teeth, as shown in Figure 1-1, are images of the first (primary) tooth. An image superelement is one that uses the geometry of another superelement to describe it for NX Nastran. These image superelements can save processing time in that they are able to use the stiffness, mass, and damping from their primary superelement, which reduces the amount of calculations needed. Full data recovery is available for image superelements. An image superelement can be an identical image, as shown in Figure 1-1, or a mirror image, as shown in Figure 1-2. In Figure 1-2, the right side of the plate is a mirror-image copy of the primary. Please note that images can have their own unique loadings. Only the stiffness, mass, and damping are identical to the primary.

Another type of superelement is the external superelement, where a part of the model is represented using matrices from an outside source. (The matrices can come from another NX Nastran run.) For these matrices, no internal geometry information is available; only the grid points to which the matrices are attached are known. An external superelement is shown in Figure 1-3. In this figure the finite element model is on the left and the external superelement is represented by the dashed lines on the right.
In static analysis, the theory used in superelement processing is exact. In dynamic analysis, approximations are made in the reduction of the mass, damping, and loading matrices. These approximations are explained in “Introduction to Superelements in Dynamic Analysis”, as are methods (in particular, component modal synthesis) to improve on the approximations.
1.3 Partitioned Solutions

When NX Nastran is processing the bulk data for a model, the input is partitioned into a separate set for each superelement, based on user instructions. The input used to accomplish this partitioning is discussed in “How to Define a Superelement”.

Once the bulk data is partitioned into separate sets, each superelement is processed individually. The degrees-of-freedom (DOFs) for each superelement are partitioned into sets in a manner identical to that used in conventional analysis. That is, all DOFs for a superelement are combined to create a G-set. Then MPCs and R-elements are used to define the M- and N-sets, and so on. (See NX Nastran User’s Guide for a complete description of sets.) The only change in the definition of sets is the definition of exterior DOFs. For each superelement, the exterior DOFs are defined as the A-set.

The exterior DOFs are best described as those that are retained for further analysis, or you can think of them as attachment DOFs, where the superelement connects to the rest of the structure. Structural matrices are assembled for each superelement, and the matrices go through reduction processing until the only remaining terms are for the A-set or attachment DOFs. These reduced matrices are used to represent the properties of the superelement when it is attached to the rest of the model.

Interior DOFs can be thought of as those that are condensed out during superelement processing.

All DOFs of a superelement that are not exterior are called interior DOFs (the omitted or O-set). These DOFs are condensed out of the matrices during the reduction process. Using either a static or dynamic reduction, the stiffness, mass, damping, and loading on these interior DOFs are transferred to the exterior DOFs.

The reduction process is best illustrated using the process known as static condensation. In this process, we start with the superelement matrices after all MPCs, R-elements, and SPCs have been processed. The set of DOFs remaining at this point are in terms of the F-set (DOFs that are not constrained), which contains the O- and A-sets as subsets. Although the interior DOFs include the M- and S-sets also, the interior DOFs in this guide are often referred to as the O-set.

Note

Each superelement is processed individually.

The static equation for the F-set is

\[ K_f U_f = P_f \]

Equation 1-1.

This equation may be rewritten as

\[
\begin{bmatrix}
K_{oo} & K_{oa} \\
-\bar{K}_{ao}^T & -\bar{K}_{aa}
\end{bmatrix}
\begin{bmatrix}
U_o \\
U_a
\end{bmatrix}
= 
\begin{bmatrix}
P_o \\
\bar{P}_a
\end{bmatrix}
\]

Equation 1-2.
where the bar over a term (for example $\bar{P}_o$) indicates that the sub-matrix represents the associated matrix of terms for that set before the reduction operation. If we look at the upper part of the equation, we obtain

$$[K_{oo}]\{U_o\} + [K_{oa}]\{U_a\} = P_o$$

Equation 1-3.

Pre-multiplying both sides of the equation by $[K_{oo}]^{-1}$ gives

$$\{U_o\} = -[K_{oo}]^{-1}[K_{oa}]\{U_a\} + [K_{oo}]^{-1}\{P_o\}$$

Equation 1-4.

At this point we will define some terms:

The static boundary transformation matrix between the exterior and interior motion is called $G_{ot}$ and is defined as

$$G_{ot} = -K_{oo}^{-1}K_{ot}$$

Equation 1-5.

Note

The ‘T’-set is a subset of the ‘A’-set. The ‘T’-set contains any “physical” exterior DOFs. In static analysis, the ‘T’-set is normally identical to the ‘A’-set. In this part of the book—when discussing static analysis—the two will often be used interchangeably.

Physically, this matrix represents the solution to the boundary motion problem. That is, each column of this matrix represents the motion of the interior points when one boundary DOF is moved one unit while the other boundary points are held constrained.

Therefore, the transformation matrix has one column for each exterior (boundary) DOF (the A-set for the superelement), and the number of rows are equal to the number of interior DOFs (the O-set for the superelement).

Also, the fixed-boundary displacements of the superelement are defined as follows:

$$U_o^o = K_{oo}^{-1}P_o$$

Equation 1-6.

This matrix represents the static solution for the displacements of the superelement when the loads are applied and the exterior points are held fixed. Based on these definitions, the total displacement of the interior points can be defined as
\[ U_o = U_o^o + G_{ot} U_t \]

Equation 1-7.

where \( U_t \) is the solution for the displacements of the exterior (boundary) points. We now substitute the equation for \( U_o \) in the lower part of Eq. 1-2, obtaining

\[ K_{ot}^T [ G_{ot} U_t + U_o^o ] + \bar{K}_{tt} U_t = \bar{P}_t \]

Equation 1-8.

From this expression we obtain the reduced stiffness and loading matrices for the superelement. The reduced stiffness, \( K_{tt} \) is defined as

\[ K_{tt} = K_{ot}^T G_{ot} + \bar{K}_{tt} \]

Equation 1-9.

and the reduced loading \( P_t \) is defined as

\[ P_t = G_{ot}^T P_o + \bar{P}_t \]

Equation 1-10.

The residual structure consists of all components of the model that were not assigned to any other superelement, plus the assembly of the reduced superelement matrices.

Each superelement is processed in this manner, and its associated matrices are reduced to the exterior DOFs. Once all superelements have been processed, the reduced matrices are assembled into a system matrix during the residual structure processing. The residual structure consists of all components of the model that were not assigned to a superelement, plus the assembly of the reduced superelement matrices.

The system or assembly solution is performed using the assembled matrices for the residual structure. Once the assembly solution is known, the boundary solution is found for each superelement. This boundary solution is used to calculate the interior displacements for each superelement, then standard data recovery is available for all superelements, including the residual structure. Any output that is available in standard (non-superelement) analysis is available in superelement analysis. The difference is that the output is now partitioned by superelement.

1.4 Example to Show the Use of Superelements in Static Analysis

The following small problem demonstrates how a static analysis is performed using superelements. The solution to the problem is first shown using conventional analysis, then using superelements.
For this example, we are looking only at motion along the axis of the points, thus the problem is simplified to contain only five DOFs.

**Conventional Analysis**

In conventional analysis, the problem is formulated in matrix form, constraints are applied, and the resulting reduced problem is solved. The five-by-five stiffness matrix is as follows:

\[
[K_{GG}] = \begin{bmatrix}
K_{12} & -K_{12} & 0 & 0 & 0 \\
-K_{12} & K_{12} + K_{23} & -K_{23} & 0 & 0 \\
0 & -K_{23} & K_{23} + K_{34} & -K_{34} & 0 \\
0 & 0 & -K_{34} & K_{34} + K_{45} & -K_{45} \\
0 & 0 & 0 & -K_{45} & K_{45}
\end{bmatrix}
\]

**Equation 1-11.**

Each row (or column) in the above matrix represents the terms associated with one DOF in the model. The terms are in ascending order, that is, the first column represents DOF 1, and the last column represents DOF 5. Replacing the springs by their numeric values, we have...
We now apply the constraints to the problem. In finite element analysis, constraints are applied by removing the associated rows and columns from the matrix; therefore, after applying constraints, we have the static equation for the constrained structure

\[
\begin{align*}
\begin{bmatrix}
U_2 \\
U_3 \\
U_4
\end{bmatrix}
&= 
\begin{bmatrix}
1 & -1 & 0 & 0 & 0 \\
-1 & 2 & -1 & 0 & 0 \\
0 & -1 & 2 & -1 & 0 \\
0 & 0 & -1 & 2 & -1 \\
0 & 0 & 0 & -1 & 1
\end{bmatrix}
^{-1}
\begin{bmatrix}
P_2 \\
P_3 \\
P_4
\end{bmatrix}
\end{align*}
\]

Equation 1-12.

or, in numeric terms

\[
\begin{align*}
\begin{bmatrix}
U_2 \\
U_3 \\
U_4
\end{bmatrix}
&= 
\begin{bmatrix}
2 & -1 & 0 \\
-1 & 2 & -1 \\
0 & -1 & 2
\end{bmatrix}
^{-1}
\begin{bmatrix}
1 \\
2 \\
3
\end{bmatrix}
\end{align*}
\]

Equation 1-13.

Solving this equation gives the solution

\[
\begin{align*}
\begin{bmatrix}
U_2 \\
U_3 \\
U_4
\end{bmatrix}
&= 
\begin{bmatrix}
2.5 \\
4.0 \\
3.5
\end{bmatrix}
\end{align*}
\]

Equation 1-15.
**Superelement Analysis**

We now formulate and solve the same problem using superelements, as shown in Figure 1-6. Because the method of defining superelements has not been discussed yet, some of what follows has not been defined. However, as you read further, more of the information will become clear.

First a flowchart showing the order of processing used to perform superelement analysis is shown in Figure 1-5.

![Flowchart for Superelement Processing](image)

*Figure 1-5. Flowchart for Superelement Processing.*
These three phases (I, II, and III) are used in all superelement solutions and will be referred to later in this guide.

![Superelement Diagram](image)

**Figure 1-6.**

The definitions of the model, as shown previously in Figure 1-6, are as follows:

- **Superelement 1 (SEID = 1)**
  - Grid points 1 and 2 are interior points. (These grid points are condensed out during the phase one operations for superelement 1.)
  - Elements $K_{12}$ and $K_{23}$ are interior or belong to superelement 1.
  - The constraint at grid point 1 is contained in superelement 1.
  - The load applied on grid point 2 is in superelement 1.
  - Grid point 3 is exterior to superelement 1. (After reduction [Phase 1] is completed for superelement 1, all that remains is a set of matrices representing the superelement attached to grid point 3.)

- **Superelement 2 (SEID = 2)**
  - Grid points 4 and 5 are interior to superelement 2.
  - Grid point 3 is exterior to superelement 2.
  - The load on grid point 4 is in superelement 2.
  - Elements $K_{34}$ and $K_{35}$ are interior to or belong to superelement 2.
  - The constraint on grid point 5 is contained in superelement 2.

- **Residual structure (R.S. OR SEID = 0)**
  - Grid point 3 is interior to the residual structure.
  - There are no elements left to belong to the residual structure.
  - The load on grid point 3 is in the residual structure.
  - Superelements 1 and 2 are processed independently, then the reduced matrices are assembled at the residual.

**Superelement 1**

After the model is divided into superelements, the data for superelement 1 contains the following information:
Based on this model, \( u_3 \) is the exterior DOF and belongs to the A-set for superelement 1. Therefore, we want to generate matrices for superelement 1, apply any constraints, and reduce the matrices to the exterior DOF. The G-set for this superelement consists of the DOFs associated with grid points 1, 2, and 3. The following are the G-sized matrices:

\[
[K_{gg}]^1 = \begin{bmatrix}
K_{12} & -K_{12} & 0 \\
-K_{12} & K_{12} + K_{23} & -K_{23} \\
0 & -K_{23} & K_{23}
\end{bmatrix}
\]

\[\{P_g\}^1 = \begin{bmatrix} P_1 \\ P_2 \\ \bar{P}_3 \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}\]

The superscript 1 shown on the matrices indicates that they belong to superelement 1. Notice that the force on grid point 3 (\( P_3 \)) is not included. Because the force is applied to an exterior point, it is not included in the superelement. This fact is indicated in the matrix for the loading by placing a bar over the \( P_3 \) term and indicating that this represents only loading on grid point 3 associated with superelement 1.

Looking at the model, we see that grid point 1 is constrained. Because that grid point is interior to the superelement, the constraint is applied as a part of the processing for superelement 1. The resulting (reduced) stiffness matrix is
This matrix is now divided into interior (O-set) and exterior (A-set) DOFs, and a static condensation is performed to reduce the matrices to the exterior DOFs.

First we compute the boundary transformation:

\[
[K_{ff}]^1 = \begin{bmatrix}
K_{12} + K_{23} & -K_{23} \\
- & - & - \\
-K_{23} & & K_{23}
\end{bmatrix}
\]

\[
= \begin{bmatrix}
K_{oo} & K_{ot} \\
- & - & - \\
K_{to} & \bar{K}_{tt}
\end{bmatrix}
\]

**Equation 1-18.**

This transformation states that if point 3 is moved 1.0 units, then point 2 will move .5 units, which is the expected result when point 1 is constrained.

Now the transformation is used to reduce the stiffness matrix to the boundary.

\[
[G_{ot}]^1 = [K_{oo}]^{-1}[K_{ot}]
\]

\[
= \frac{K_{23}}{K_{12} + K_{23}} = 0.5
\]

**Equation 1-19.**

The results make sense. Superelement 1 contains two springs in series, and the resulting stiffness as seen at grid point 3 is 0.5.

Now we have to reduce the applied loadings to the boundary. After applying the constraint to the loading matrix, we have
\[ \begin{bmatrix} P_2 \\ \bar{P}_3 \end{bmatrix}^1 = \begin{bmatrix} 1 \\ 0 \end{bmatrix} \]

\[ \begin{cases} P_3 \\ \bar{P}_t \end{cases} \]

Equation 1-21.

which can be reduced to the boundary as follows:

\[ \begin{bmatrix} P_t \end{bmatrix}^1 = \begin{bmatrix} \bar{P}_t + G_{ot}^T P_o \end{bmatrix} \]

\[ P_3^1 = \frac{K_{23}}{K_{12} + K_{23}} P_2 = 0.5 \]

Equation 1-22.

Once again, the result is what we would expect. If grid point 3 is constrained and the unit force is applied to grid point 2, the reaction seen at point 3 is 0.5 units.

**Superelement 2**

The data for superelement 2 contains the following information

\[ u_3 \text{ is the exterior DOF and belongs to the A-set for superelement 2. Therefore, we want to generate matrices for superelement 2, apply any constraints, and reduce the matrices to the exterior DOF.} \]
$$[K_{gg}]^2 = \begin{bmatrix}K_{34} & -K_{34} & 0 \\ -K_{34} & K_{34} + K_{45} & -K_{45} \\ -K_{34} & -K_{45} & K_{45}\end{bmatrix}$$

**Equation 1-23.**

\[
\{P_g\}^2 = \begin{bmatrix}\bar{P}_3^2 \\ P_4 \\ P_5\end{bmatrix} = \begin{bmatrix}0 \\ 3 \\ 0\end{bmatrix}
\]

**Equation 1-24.**

Notice that the force on grid point 3 does not show up in superelement 2 either. Once again, forces on exterior points are not included in the superelement matrices.

The constraint will be applied, this time at DOF5, then the boundary transformation will be calculated and applied, resulting in the following:

\[
\begin{align*}
[G_{ot}]^2 &= \frac{K_{34}}{K_{34} + K_{45}} = 0.5 \\
[K_{tt}]^2 &= \frac{K_{34}K_{45}}{K_{34} + K_{45}} = 0.5 \\
0 &= \frac{K_{34}}{K_{34} + K_{45}} = 1.5 \\
\end{align*}
\]

**Equation 1-25.**

The transformation and reduced matrices make sense. If grid point 3 is moved 1.0 unit, grid point 4 will move 0.5 units. As before, the stiffness is two springs in series, resulting in a combined stiffness of 0.5, and the load of 3.0 units at grid point 4 gives a 1.5 unit reaction at point 3 if it is constrained.

**Residual Structure**

The remaining structure, or grid point 3, is defined as the residual structure. Matrices are generated for the residual structure, based on any elements or loads remaining, then the reduced matrices from the
superelements are added at the appropriate DOFs. Once the combined assembly matrices exist, any constraints applicable to the remaining DOFs are applied and the residual structure problem is solved.

Figure 1-9 shows the residual structure. The individual components that are assembled to make up the residual are shown on the left. The resulting assembly model is shown on the right.

The residual structure for this model contains no elements, only one grid point, the physical load on that point, and the reduced matrices from the superelements.

Because all physical constraints have been applied at the superelement level, no reduction is performed at the residual level for this model. If there were a physical model for the residual, then it would also go through the application of constraints and a reduction to a final set of analysis matrices. Therefore, the assembly matrix is the result of adding the superelement matrices together at grid point 3, or

\[
\begin{align*}
[K_{gg}] &= [K_{aa}] = [K_{aa}^1 + K_{aa}^2 + K_{gg}^0] \\
K &= K^1 + K^2 = 1
\end{align*}
\]

Equation 1-26.

where the matrices $K_{aa}^1$ and $K_{aa}^2$ represent the reduced superelement stiffness matrices, and the matrix $K_{gg}^0$ represents the stiffness matrix resulting from any elements in the residual structure. In this problem there are no elements in the residual structure; therefore, $K_{gg}^0$ is null.

Similarly, the loading matrix is the physical loadings applied on the residual, plus the reduced superelement loads. The 2 unit force on grid point 3 is finally included at this point, because it is acting on a grid point that is interior to the residual structure.

\[
\{P_g\} = \{P_a\} = \{P_a^1 + P_a^2 + P_g^0\}
\]

Equation 1-27.
\[ P = P_3^1 + P_3^2 + P_3^0 = 4 \]

Equation 1-28.

Now that the stiffness and loading matrices have been generated and reduced, we are ready to solve the residual structure problem for the A-set displacements:

\[ \{ U_a \} = [ K_{tt} ]^{-1} \{ P_t \} \]

Equation 1-29.

\[ U_3^0 = \frac{P}{K} = 4 \]

Equation 1-30.

We now have the displacement solution for the residual structure, and we are ready to begin data recovery. Data recovery is processed for each superelement independently, allowing segmented or selective data recovery.

**Superelement 1**

We already have the solution for the exterior point (grid point 3), and now we are ready to perform data recovery for the interior points of superelement 1. First, we calculate the fixed-boundary solution. That is, we calculate the motion of superelement 1 if the exterior point is held constrained and the loadings are applied.

\[ \left\{ U_o^{0} \right\} = [ K_{oo} ]^{-1} \{ P_o \} \]

\[ U_2^0 = \frac{1}{K_{12} + K_{23}} P_2 = 0.5 \]

\[ P_2 = 1 \]

\[ u_2^0 = ? \]

Figure 1-10.

Now we calculate the solution to the boundary motion.
Finally, by superposition the interior displacement is

\[ \{ \begin{align*} u_2^i &= \begin{cases} \ast & \text{if} \ \text{known} \\ \ast & \text{if} \ \text{unknown} \end{cases} \\ u_3 &= 4 \end{align*} \] \[ \begin{align*} \{ u'_2 \} &= \left[ G_{o1} \right] \cdot U_3 = 2.0 \end{align*} \]

which is identical to the solution obtained for grid point 2 when the model was solved without using superelements. Element output, SPC forces, and all other standard output can now be calculated for superelement 1.

**Superelement 2**

Once again, the boundary solution is known from the residual structure solution, and data recovery is similar to that performed on superelement 1. Data recovery is not shown for superelement 2 but also gives answers identical to the solution obtained without superelements.

Simply stated, a static superelement solution is the combination of the solution obtained by adding the fixed-boundary solution to the solution obtained by imposing the boundary displacements.

In static analysis, there are no approximations made using superelements.

**Sample Problem**

For consistency, throughout this guide the model shown in Figure 1-12 is used as a sample problem to demonstrate NX Nastran input and output. The model, best described as a metal stamping (also known as a two-headed flyswatter), has absolutely no resemblance to any real problem. The reasons for this choice are as follows: Most real models are much too complicated to be used as a simple example (the input is too cumbersome), and a model that might be reasonable to people working in one industry may have no meaning whatsoever to someone in another industry. The model is simple and needs no description.

The bulk data for the model is shown in Listing 1-1. This listing uses replicators to generate many of the entries and is not shown again in this guide. In subsequent chapters and examples, this listing is
referred to using the INCLUDE entry. In this way, listings shown in this guide contain only information specific for that run, resulting in less confusion.

**Figure 1-12. Two-Headed Flyswatter.**

The structure is made of steel with Young’s modulus of 30,000,000 psi, weight density of 0.283 lb/in.\(^3\), and Poisson’s ratio of 0.3. Grid points 1 and 2 are constrained. In static analysis, the structure will undergo three loadings:

- Pressure applied on elements 18 through 67.
- Point loads of two pounds acting in the positive Z direction at grid points 93 and 104.
- Opposing two pound point loads in the Z direction at grid points 93 and 104.
Table 1-1. Listing of Bulk Data for the Sample Problem.

```
$  
$  *******************************************************
$  $  BASIC MODEL DEFINITION - SAME FOR ALL RUNS
$  *******************************************************
$  
$  FILE NAME IS MODEL.DAT
$  
GRDSET,,,,,,,6
GRID,1,-.4,0.,0.,123456
GRID,3,-.4,0.9,0.
  =,*2,=,=*/.9,==
 =1
GRID,2,.4,0.0.,123456
GRID,4,.4,0.9,0.
  =,*2,=,=*/.9,==
 =1
GRID,9,-3.6,3.6,0.
  =1,=*/.8,==
 =8
GRID,19,-3.6,4.4,0.
  =1,=*/.8,==
 =8
GRID,29,-3.6,5.2,0.
GRID,30,-2.8,5.2,0.
GRID,31,2.8,5.2,0.
GRID,32,3.6,5.2,0.
GRID,33,-5.2,6.,0.
  =1,=*/.8,==
 =4
GRID,39,1.2,6.,0.
  =1,=*/.8,==
 =4
GRID,45,-5.2,6.8,0.
  =1,=*/.8,==
 =4
GRID,51,1.2,6.8,0.
  =1,=*/.8,==
 =4
GRID,57,-5.2,7.6,0.
  =1,=*/.8,==
 =4
GRID,63,1.2,7.6,0.
  =1,=*/.8,==
 =4
GRID,69,-5.2,8.4,0.
  =1,=*/.8,==
 =4
GRID,75,1.2,8.4,0.
  =1,=*/.8,==
 =4
GRID,81,-5.2,9.2,0.
  =1,=*/.8,==
 =4
GRID,87,1.2,9.2,0.
  =1,=*/.8,==
 =4
GRID,93,-5.2,10.,0.
  =1,=*/.8,==
 =4
GRID,99,1.2,10.,0.
  =1,=*/.8,==
 =4
$  
$  ELEMENTS
```

---

It appears to be a listing of bulk data for a sample problem, possibly related to finite element analysis. The data includes grid definitions and element data. The file name is 'MODEL.DAT'. Each grid definition specifies a set of coordinates and dimensions, and each element definition specifies a set of nodes and their connectivity.
Table 1-1. Listing of Bulk Data for the Sample Problem.

```
CQUAD4, 1, 1, 1, 2, 4, 3
=*1, *=2, *=2, *=2
=1
CQUAD4, 4, 1, 7, 8, 14, 13
=*1, *=1, *=1, *=1, *=1
=2
CQUAD4, 5, 1, 13, 14, 24, 23
=*1, *=1, *=1, *=1
=2
CQUAD4, 14, 1, 19, 20, 30, 29
CQUAD4, 15, 1, 29, 30, 36, 35
CQUAD4, 16, 1, 27, 28, 32, 31
CQUAD4, 17, 1, 31, 32, 42, 41
CQUAD4, 18, 1, 33, 34, 46, 45
=*1, *=1, *=1, *=1, *=1
=3
CQUAD4, 23, 1, 45, 46, 58, 57
=*1, *=1, *=1, *=1, *=1
=3
CQUAD4, 28, 1, 57, 58, 70, 69
=*1, *=1, *=1, *=1, *=1
=3
CQUAD4, 33, 1, 69, 70, 82, 81
=*1, *=1, *=1, *=1, *=1
=3
CQUAD4, 38, 1, 81, 82, 94, 93
=*1, *=1, *=1, *=1, *=1
=3
CQUAD4, 43, 1, 39, 40, 52, 51
=*1, *=1, *=1, *=1, *=1
=3
CQUAD4, 48, 1, 51, 52, 64, 63
=*1, *=1, *=1, *=1, *=1
=3
CQUAD4, 53, 1, 63, 64, 76, 75
=*1, *=1, *=1, *=1, *=1
=3
CQUAD4, 58, 1, 75, 76, 88, 87
=*1, *=1, *=1, *=1, *=1
=3
CQUAD4, 63, 1, 87, 88, 100, 99
=*1, *=1, *=1, *=1, *=1
=3
MAT1, 1, 30.6, .3, .283
PARAM, WTMAS, .00259
PARAM, AUTOSPC, YES
PSHELL, 1, 1, .05, 1, , 1
```

Chapter 2: How to Define a Superelement

- Defining a Superelement Using Partitioned Bulk Data (PARTs)
- Defining Superelements in the Main Bulk Data Section
- Defining External Superelements
Now that the basic concept of superelements has been explained, it is time to learn how to define superelements in NX Nastran. Superelements are defined using the Bulk Data Section of the input file. There are two methods available for defining superelements: main bulk data superelements and PARTs.

The following sections provide a description of each method, with examples. Each superelement in NX Nastran is identified by an integer identification known as the SEID. Each SEID must be a unique positive integer (with the exception of the residual structure, which is known as superelement 0).

If no superelements are defined, the model is assumed to be a residual-structure-only model, and a conventional (non-superelement) solution is performed. By default, all superelement solutions perform a conventional solution if no superelements are defined.

As the name implies, main bulk data superelements are defined in the Main Bulk Data Section of the input file. When superelements are defined using this approach, the model defined in this section of the input is divided into separate components. (Each component is a superelement.) A good way to describe this is to say that the program is using a cookie-cutter approach with the model, taking a complete model and dividing it into superelements.

PART superelements are defined in a different manner. Each superelement is defined in its own Partitioned Bulk Data Section. These separate sections of the bulk data are self-contained in that each section contains all geometry, elements, properties, constraints, and loading data for that component of the model. When PARTs are used, the program works in a manner similar to an assembly process. That is, a series of separate components are assembled into the final finite element model.

An external superelement is a reduced boundary representation of a full component model stored in an external file such as an .op2, .op4, bulk data, or database. After you create several external superelements in specific “creation” solutions, you can then combine them in a full NX Nastran system analysis.

All of these approaches can be used independently or together.

### 2.1 Defining a Superelement Using Partitioned Bulk Data

#### Defining PARTs

PARTs are defined as separate components using separate areas of the Bulk Data Section. Therefore, each PART can be thought of as a separate component model. NX Nastran automatically locates any coincident grid points in the PARTs and connects the component models to create the assembly model.

The Bulk Data Section can be divided into separate sections for each PART. This division is accomplished by using the BEGIN SUPER entry. The format of this entry is as follows:

```
BEGIN [BULK] SUPER = i
```

where \( i \) is the superelement id being defined. The commonly used form of this command is:

```
BEGIN SUPER = i
```

which is the form used in this book.
Prior to MSC.Nastran Version 69, the Bulk Data Section of the input file was a single section of data with the exception of auxmodels in optimization, which contained the complete model definition. The entire model was defined in the area between BEGIN BULK and ENDDATA. Each grid point had to be unique, and each element ID had to be unique. In MSC.Nastran Version 69, it became possible to partition the Bulk Data Section of the input into separate component models, using the BEGIN SUPER command. Thus, each of these component models is a self-contained model defining a PART of the total model. Within each of these sections, grid point and element IDs must be unique as before; however, different PARTs can reuse grid and element IDs, because the sections are separate in the input file.

The Bulk Data Section Using PARTs

When PARTs are used, the bulk data is divided into different sections. The section of the bulk data contained between the BEGIN BULK and either the first BEGIN SUPER or the ENDDATA is known as the Main Bulk Data Section. If only this area is used, then the file is compatible with versions of MSC.Nastran prior to Version 69. In this section, superelements can be defined (although it is not required) using the method shown in "Defining Superelements in the Main Bulk Data Section." Any superelements defined in this section are known as main bulk data superelements.

When you define PARTs, it is not necessary for you to tell NX Nastran where to make the connections to other superelements. The program contains logic to determine which grid points are coincident and automatically connects (by default) any coincident grid points. Later in this section, we discuss how to override the automatic connection.

Format of the Input File When PARTs Are Used

When PARTs are used, the Executive and Case Control Sections are unchanged; only the Bulk Data Section is different. A sample input file looks like the following:

```text
SOL 101
CEND
TITLE = Sample Input File Demonstrating PARTs
.
.
BEGIN BULK
$
$ MAIN BULK DATA SECTION
$
$
BEGIN SUPER = 1
$
$ data for PART 1
$
$
BEGIN SUPER = 25
$
$ data for PART 25
$
$
ENDDATA
```

In this example, there is a Main Bulk Data Section (in which you can define some main bulk data superelements) and two PART superelements (1 and 25). Each section is self-contained. That is, no entry in PART 1 can reference an entry in any other section of the input. This goes for all PARTs; they
must be self-contained. There are several main bulk data entries that can be used to move, copy, or manually connect PARTs, but beyond these entries, no entry in any section of the input can reference an entry in any other section of the bulk data.

We will look at how to define the two-headed flyswatter model (from the previous section) using PARTs. We are going to define the model using seven PART superelements and a residual structure. The figure below shows how the model will be divided into superelements.

![Diagram of superelements](image)

**Figure 2-1. Residual Structure for the Sample Problem Divided into Superelements.**

Each superelement in this model is defined using a PART or a separate section of the bulk data. Therefore, there are seven separate BEGIN SUPER sections in our input file. The residual structure is defined in the main bulk data. (This is not required; we are simply showing that this is an option.)

The following input file is used to solve this problem:

```plaintext
$ file - sels10lp.dat
$ all 7 s.e. brought in using begin super
$ duplicate boundary grid ids
$ each s.e. contains its own property description
```
\$ SOL 101
CEND
TITLE = S.E. SAMPLE PROBLEM 1
SUBTITLE = S.E. STATICS - RUN 1 - MULTIPLE LOADS
DISP = ALL
\$
\$ default is super = all for V69
\$
\$ set defaults for all se - see section 5
\$
PARAM,GRDPNT,0
PARAM,WTMASS,.00259
SUBCASE 101
LABEL = PRESSURE LOAD
LOAD = 101
\$
SUBCASE 201
LABEL = 2# NORMAL LOADS
LOAD = 201
\$
SUBCASE 301
LABEL = OPPOSING LOADS
LOAD = 301
\$
include 'plot.dat'
\$
BEGIN BULK
\$
\$ main bulk data section
\$
include 'part0.dat'
\$
begin super=1
\$
include 'loadprt1.dat'
include 'part1.dat'
\$
begin super=2
\$
include 'loadprt2.dat'
include 'part2.dat'
\$
begin super=3
\$
include 'part3.dat'
\$
begin super=4
\$
include 'part4.dat'
\$
begin super=5
\$
include 'part5.dat'
\$
begin super=6
\$
include 'part6.dat'
\$
begin super=7
\$
include 'part7.dat'
\$
enddata
The listings of the included files are given below. The executive control selects SOL 101, which is the static solution that includes superelements. The case control defines three solutions for this model and provides default parameter values for all superelements in the model (a detailed description of the case control for superelements is in Section 5). The file plot.dat is a file requesting plots and is not shown until the section describing superelement plotting.

The Main Bulk Data Section of the file is very short; for this model it contains the physical model of the residual structure. This model is defined in the file part0.dat, which is shown here:

```
CQUAD4 5 1 13 14 24 23
$ GRDSET 6
GRID 13 -.4 3.6 0.
GRID 14 .4 3.6 0.
GRID 23 -.4 4.4 0.
GRID 24 .4 4.4 0.
$ MAT1,1,30.+6./.3,.283
PSHELL,1,1,.05,1,,1
$ param,witmass,.002588
```

This file contains the physical model of the residual structure, which consists of element 5 and the grid points to which it is attached, along with the associated properties and parameters. Notice that this file contains only the above information and does not contain copies of the grid points that are used to make the connection between PARTs. (You can place copies of these points in the residual structure if you wish—the answers are the same for single-level solutions). The model of the residual structure ends with the end of the Main Bulk Data Section or at the BEGIN SUPER = 1. Subsequent entries belong to the PARTs.

The section of the input file defining PART 1 is next and contains two include statements. These statements insert two files, loadprt1.dat (loading data) and part1.dat (the model of the PART), into the runstream. These two files follow.

```
$ file - loadprt1.dat
$ loads on s.e. 1
$ LOAD CASE 1 - PRESSURE LOAD
$ PLOAD2,101,-1.,18,THRU,42
$ LOAD CASE 2 - 2 POINT LOADS AT CORNERS
$ FORCE,201,93,,2.,0.,0.,1.
$ LOAD CASE 3 - OPPOSING POINT LOADS AT CORNERS
$ FORCE,301,93,,2.,0.,0.,1.
$ File loadprt1.dat contains the data that defines the loadings to be applied to PART 1 for the three static solutions.
```

```
CQUAD4 18 1 33 34 46 45
CQUAD4 19 1 34 35 47 46
CQUAD4 20 1 35 36 48 47
CQUAD4 21 1 36 37 49 48
CQUAD4 22 1 37 38 50 49
CQUAD4 23 1 45 46 58 57
```
<table>
<thead>
<tr>
<th>Element Type</th>
<th>Number</th>
<th>Coordinates</th>
</tr>
</thead>
<tbody>
<tr>
<td>CQUAD4</td>
<td>24</td>
<td>46, 47, 59, 58</td>
</tr>
<tr>
<td>CQUAD4</td>
<td>25</td>
<td>47, 48, 60, 59</td>
</tr>
<tr>
<td>CQUAD4</td>
<td>26</td>
<td>48, 49, 61, 60</td>
</tr>
<tr>
<td>CQUAD4</td>
<td>27</td>
<td>49, 50, 62, 61</td>
</tr>
<tr>
<td>CQUAD4</td>
<td>28</td>
<td>57, 58, 70, 69</td>
</tr>
<tr>
<td>CQUAD4</td>
<td>29</td>
<td>58, 59, 71, 70</td>
</tr>
<tr>
<td>CQUAD4</td>
<td>30</td>
<td>59, 60, 72, 71</td>
</tr>
<tr>
<td>CQUAD4</td>
<td>31</td>
<td>60, 61, 73, 72</td>
</tr>
<tr>
<td>CQUAD4</td>
<td>32</td>
<td>61, 62, 74, 73</td>
</tr>
<tr>
<td>CQUAD4</td>
<td>33</td>
<td>69, 70, 82, 81</td>
</tr>
<tr>
<td>CQUAD4</td>
<td>34</td>
<td>70, 71, 83, 82</td>
</tr>
<tr>
<td>CQUAD4</td>
<td>35</td>
<td>71, 72, 84, 83</td>
</tr>
<tr>
<td>CQUAD4</td>
<td>36</td>
<td>72, 73, 85, 84</td>
</tr>
<tr>
<td>CQUAD4</td>
<td>37</td>
<td>73, 74, 86, 85</td>
</tr>
<tr>
<td>CQUAD4</td>
<td>38</td>
<td>81, 82, 94, 93</td>
</tr>
<tr>
<td>CQUAD4</td>
<td>39</td>
<td>82, 83, 95, 94</td>
</tr>
<tr>
<td>CQUAD4</td>
<td>40</td>
<td>83, 84, 96, 95</td>
</tr>
<tr>
<td>CQUAD4</td>
<td>41</td>
<td>84, 85, 97, 96</td>
</tr>
<tr>
<td>CQUAD4</td>
<td>42</td>
<td>85, 86, 98, 97</td>
</tr>
</tbody>
</table>

GRDSET 6
boundary grids
GRID 35 -3.6 6.0 0.
GRID 36 -2.8 6.0 0.
GRID 33 -5.2 6.0 0.
GRID 34 -4.4 6.0 0.
GRID 37 -2.0 6.0 0.
GRID 38 -1.2 6.0 0.
GRID 45 -5.2 6.8 0.
GRID 46 -4.4 6.8 0.
GRID 47 -3.6 6.8 0.
GRID 48 -2.8 6.8 0.
GRID 49 -2.0 6.8 0.
GRID 50 -1.2 6.8 0.
GRID 57 -5.2 7.6 0.
GRID 58 -4.4 7.6 0.
GRID 59 -3.6 7.6 0.
GRID 60 -2.8 7.6 0.
GRID 61 -2.0 7.6 0.
GRID 62 -1.2 7.6 0.
GRID 69 -5.2 8.4 0.
GRID 70 -4.4 8.4 0.
GRID 71 -3.6 8.4 0.
GRID 72 -2.8 8.4 0.
GRID 73 -2.0 8.4 0.
GRID 74 -1.2 8.4 0.
GRID 81 -5.2 9.2 0.
GRID 82 -4.4 9.2 0.
GRID 83 -3.6 9.2 0.
GRID 84 -2.8 9.2 0.
GRID 85 -2.0 9.2 0.
GRID 86 -1.2 9.2 0.
GRID 93 -5.2 10.0 0.
GRID 94 -4.4 10.0 0.
GRID 95 -3.6 10.0 0.
GRID 96 -2.8 10.0 0.
GRID 97 -2.0 10.0 0.
GRID 98 -1.2 10.0 0.
MAT1,1,30.,+6,,.3,.283
PARAM,WTMASS,.00259
PARAM,AUTOSPC,YES
File part1.dat contains the physical model for PART1. If you look at the illustration of the model provided in the previous section, you will notice that this PART attaches to PART 3 at grid points 35 and 36. Notice that these points are contained in the data for this component of the model.

PART 2 is defined by similar files (shown later in this section) and will not be discussed here.

PART 3 is defined by a single file, part3.dat:

```
PSHELL,1,1,.05,1,,1
$

CQUAD4 14 1 19 20 30 29
CQUAD4 15 1 29 30 36 35
$
GRDSET 6
$
boundary grids
$
GRID 19 -3.6 4.4 0.
GRID 20 -2.8 4.4 0.
GRID 35 -3.6 6. 0.
GRID 36 -2.8 6. 0.
$
GRID 29 -3.6 5.2 0.
GRID 30 -2.8 5.2 0.
$
MAT1,1,30.+6.,.3.,.283
PARAM,WTMASS,.00259
PARAM,AUTOSPC,YES
PSHELL,1,1,.05,1,,1
$
```

This file contains the physical model for PART 3. This PART connects to PART 1 at grid points 35 and 36 (both included in this section) and also connects to PART 5 at grid points 19 and 20 (also included in this section). When processing PARTs, NX Nastran looks for grid points that are coincident, such as grid points 35 and 36 in PART 1 and grid points 35 and 36 in PART 3. Unless instructed otherwise, the program connects these components at these points. Therefore, grid points 35 and 36 are exterior to superelement 1. There is no requirement that the grid points have the same ID in different PARTs, only that they be coincident within a tolerance (described later). When coincident points are found, the program notes that they should be connected and (unless there is a coincident point in the residual structure model or you are using a multi-level tree) creates an internal grid point in the residual structure model that is coincident with these points. The matrices for each component are reduced to the external DOFs and passed downstream to the residual structure, where the matrices are combined and solved.

All boundary grids are identified in the output from NX Nastran (see output from this sample in Section 9) and listed as exterior (or interior) to the appropriate superelements. Once all PARTs have been read and all coincident grid points have been found, the program is ready to process the PARTs.

The following files are the rest of the input for this example.

```
CQUAD4 43 1 39 40 52 51
CQUAD4 44 1 40 41 53 52
CQUAD4 45 1 41 42 54 53
CQUAD4 46 1 42 43 55 54
CQUAD4 47 1 43 44 56 55
```
CQUAD4  48  1  51  52  64  63  
CQUAD4  49  1  52  53  65  64  
CQUAD4  50  1  53  54  66  65  
CQUAD4  51  1  54  55  67  66  
CQUAD4  52  1  55  56  68  67  
CQUAD4  53  1  63  64  76  75  
CQUAD4  54  1  64  65  77  76  
CQUAD4  55  1  65  66  78  77  
CQUAD4  56  1  66  67  79  78  
CQUAD4  57  1  67  68  80  79  
CQUAD4  58  1  75  76  88  87  
CQUAD4  59  1  76  77  89  88  
CQUAD4  60  1  77  78  90  89  
CQUAD4  61  1  78  79  91  90  
CQUAD4  62  1  79  80  92  91  
CQUAD4  63  1  87  88  100  99  
CQUAD4  64  1  88  89  101  100  
CQUAD4  65  1  89  90  102  101  
CQUAD4  66  1  90  91  103  102  
CQUAD4  67  1  91  92  104  103  
$  
GRDSET  6  
$  
$ boundary grids  
$  
GRID  41  2.8  6.  0.  
GRID  42  3.6  6.  0.  
$  
GRID  39  1.2  6.  0.  
GRID  40  2.  6.  0.  
GRID  43  4.4  6.  0.  
GRID  44  5.2  6.  0.  
$  
GRID  51  1.2  6.8  0.  
GRID  52  2.  6.8  0.  
GRID  53  2.8  6.8  0.  
GRID  54  3.6  6.8  0.  
GRID  55  4.4  6.8  0.  
GRID  56  5.2  6.8  0.  
GRID  63  1.2  7.6  0.  
GRID  64  2.  7.6  0.  
GRID  65  2.8  7.6  0.  
GRID  66  3.6  7.6  0.  
GRID  67  4.4  7.6  0.  
GRID  68  5.2  7.6  0.  
GRID  75  1.2  8.4  0.  
GRID  76  2.  8.4  0.  
GRID  77  2.8  8.4  0.  
GRID  78  3.6  8.4  0.  
GRID  79  4.4  8.4  0.  
GRID  80  5.2  8.4  0.  
GRID  87  1.2  9.2  0.  
GRID  88  2.  9.2  0.  
GRID  89  2.8  9.2  0.  
GRID  90  3.6  9.2  0.  
GRID  91  4.4  9.2  0.  
GRID  92  5.2  9.2  0.  
GRID  99  1.2  10.  0.  
GRID  100  2.  10.  0.  
GRID  101  2.8  10.  0.  
GRID  102  3.6  10.  0.  
GRID  103  4.4  10.  0.  
GRID  104  5.2  10.  0.  
$  
MAT1,1,30.+6,,.3,.283  
PARAM,WTMASS,.00259
PARAM, AUTOSPC, YES
PSHELL, 1, 1, .05, 1, 1
$
$
$ file - loadprt2.dat
$ loads on s.e. 2
$
$ LOAD CASE 1 - PRESSURE LOAD
$ PLOAD2, 101, -1., 43, THRU, 67
$
$ LOAD CASE 2 - 2 POINT LOADS AT CORNERS
$ FORCE, 201, 104, 2., 0., 0., 1.
$
$ LOAD CASE 3 - OPPOSING POINT LOADS AT CORNERS
$ FORCE, 301, 104, 2., 0., 0., -1.
$
$
$ part4.dat
$
CQUAD4 16 1 27 28 32 31
CQUAD4 17 1 31 32 42 41
$
GRDSET 6
$
$ boundary grids
$
GRID 27 2.8 4.4 0.
GRID 28 3.6 4.4 0.
GRID 41 2.8 6. 0.
GRID 42 3.6 6. 0.
$
GRID 31 2.8 5.2 0.
GRID 32 3.6 5.2 0.
$
MAT1, 1, 30.+6, .3, .283
PARAM, WTMASST, .00259
PARAM, AUTOSPC, YES
PSHELL, 1, 1, .05, 1, 1
$
$
$ part5.dat
$
CQUAD4 6 1 9 10 20 19
CQUAD4 7 1 10 11 21 20
CQUAD4 8 1 11 12 22 21
CQUAD4 9 1 12 13 23 22
$
GRDSET 6
$
$ boundary grids
$
GRID 19 -3.6 4.4 0.
GRID 20 -2.8 4.4 0.
GRID 13 -.4 3.6 0.
GRID 23 -.4 4.4 0.
$
GRID 9 -3.6 3.6 0.
GRID 10 -2.8 3.6 0.
GRID 11 -2. 3.6 0.
GRID 12 -1.2 3.6 0.
GRID 21 -2. 4.4 0.
Automatically Connecting PARTs to Other Components of the Model

By default, the program automatically connects points from a PART to any coincident points in any other PARTs or in the Main Bulk Data Section. There is no need to be concerned with coordinate systems on these coincident points; NX Nastran automatically connects coincident points, accounting for differences in the output coordinate systems.

These points are identified as boundary points in the output from NX Nastran. By default, no special effort is required from you to make the connection. If for one connection all the boundary points belong to PARTs (none of the coincident points are in the residual structure), then NX Nastran creates a new point internally that is coincident with the boundary points and places that internal point in the residual structure (or the lowest connected superelement in a multi-level tree). These internal points can not be constrained. If you wish to apply a constraint on this point, you can define a coincident grid point in the residual structure (or in the lowest connected superelement for a multi-level tree) and constrain that point, or you can apply constraints in the PARTs (subject to the limitations in the next paragraph).

Constraints on Connecting Points

If you need to apply constraints on the points where two or more PARTs connect, there are limitations. If you wish to constrain all six DOFs, there is no limitation; place the constraint on the connecting grid point in any of the PARTs and all other coincident points are also constrained in all six DOFs. If the constraint is not on all six DOFs, then you must exercise care. Because the program will allow the connection of coincident points that have different output coordinate systems, it would be difficult to allow constraints on one of the points in one coordinate system and to correctly map that set of constraints onto another system. Therefore, the following set of rules is enforced at connecting points:

• If the PARTs being connected are at the same level in the superelement tree, either all six or none of the DOFs must be constrained.

• If one of the PARTs is at a lower level in the tree, then you must apply the desired constraints at the connection inside the PART that is lowest in the processing tree (last of the group to be processed).

Controlling the Connection Between PARTs and the Rest of the Model

The SEBNDRY, SEBULK, SECONCT, and SEEXCLD entries can appear only in the Main Bulk Data Section of your input file.

If you want to override the automatic connection, there are several options available to you. You can control this operation by using the SEBNDRY, SEBULK, SECONCT, and SEEXCLD entries in the Main Bulk Data Section of your input file.

Although a more thorough explanation follows, here is a quick definition of each of these entries:

SEBNDRY Provides a list of grid points that can be connected between a PART and one or more PARTs (used to limit the automatic search for coincident points).

SEBULK Defines boundary search options (sets tolerance for coincident grid point checks).
SECONCT  Explicitly defines the GRIDs and SPOINTs to be connected between PARTs (overrides automatic search logic) and allows you to set the tolerance for the coincident point test.

SEEXCLD  Provides a list of points in a PART that cannot be connected to one or more other PARTs (used to limit automatic search logic).

The default tolerance in the coincident grid search logic is 1.0E-5 units.

By default, NX Nastran uses the automatic logic to find coincident points between a PART and the other superelements in a model. The default tolerance on the search is 1.0E-5 units. This default can be changed by using either the SEBULK or SECONCT entry.

SEBNDRY

The format of the SEBNDRY entry follows:

Format:

<table>
<thead>
<tr>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
</tr>
</thead>
<tbody>
<tr>
<td>SEBNDRY</td>
<td>SEIDA</td>
<td>SEIDB</td>
<td>GIDA1</td>
<td>GIDA2</td>
<td>GIDA3</td>
<td>GIDA4</td>
<td>GIDA5</td>
<td>GIDA6</td>
<td></td>
</tr>
<tr>
<td>GIDA7</td>
<td>GIDA8</td>
<td>-etc.-</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

This entry is used to limit the automatic search logic to selected grid points. Any grid points listed on this entry are the only grid points in SEIDA to which the automatic logic can connect grid points in SEIDB.

Description of the fields on this entry:

SEIDA  PART for which this entry is used in the automatic search routines.

SEIDB  PART(s) for which the search logic uses this list when searching for points coincident to the ones in SEIDA, listed here. This field either contains an integer superelement ID or the word ALL, if the list is used for all superelements when searching for coincident points for SEIDA.

GIDAi  Grid points in SEIDA that can be connected to SEIDB if there are coincident points in SEIDB.

Example 1:

<table>
<thead>
<tr>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
</tr>
</thead>
<tbody>
<tr>
<td>SEBNDRY</td>
<td>400</td>
<td>4</td>
<td>10</td>
<td>20</td>
<td>30</td>
<td>40</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

This entry states that when searching for grid points in superelement 4 that are coincident to points in PART 400, only grid points 10, 20, 30, and 40 in PART 400 can be used. No other grid points in superelement 400 can be connected to points in superelement 4, even if they are coincident.
**SEBULK**

The format of the **SEBULK** entry is as follows:

**Format:**

<table>
<thead>
<tr>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
</tr>
</thead>
<tbody>
<tr>
<td>SEBULK</td>
<td>SEID</td>
<td>TYPE</td>
<td>RSEID</td>
<td>METHOD</td>
<td>TOL</td>
<td>LOC</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

This entry has a number of uses. For purposes of the current section, we only discuss using it to control the automatic search logic for coincident grid points.

In this context, a description of the fields on this entry follows:

- **SEID**: Superelement number for which this SEBULK entry is being used. You can have SEBULK entries for each PART in your model.

- **TYPE**: There are several TYPEs allowed. For purposes of the current discussion, only PRIMARY is considered. (The other TYPEs involve more advanced features.)—no default value.

- **RSEID**: Reference superelement ID—also an advanced feature to be discussed later.

- **METHOD**: Boundary GRID point search method—can be AUTO (default) or MANUAL. If this is MANUAL, then SECONCT entries must be used for this PART to make the connections to the rest of the model.

- **TOL**: Location tolerance for coincident grid point test (default = 1.E-5).

- **LOC**: Coincident grid point test option for MANUAL connection. Can be YES (default = grid points must be within TOL, or a FATAL will be issued) or NO. (Do not test for coincidence—may be dangerous).

**Example:**

<table>
<thead>
<tr>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
</tr>
</thead>
<tbody>
<tr>
<td>SEBULK</td>
<td>14</td>
<td>PRIMARY</td>
<td>AUTO</td>
<td>1.0E-3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

This entry instructs NX Nastran to use the automatic coincident grid point search logic to find the attachment points for superelement 14, but to use a tolerance of 1.0E-3 units.

**SECONCT**

The format of the **SECONCT** entry follows:

**Format:**

<table>
<thead>
<tr>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
</tr>
</thead>
<tbody>
<tr>
<td>SECONCT</td>
<td>SEIDA</td>
<td>SEIDB</td>
<td>TOL</td>
<td>LOC</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
This entry manually specifies points (GRID or SPOINT) to be connected between two superelements. If this entry is used and METHOD=AUTO on the SEBULK entry, these points are added to the list of points found by the automatic logic. If METHOD=MANUAL on the SEBULK, only these points are connected between these two superelements.

The fields on this entry are used as follows:

SEIDA | PART to be connected.

SEIDB | Superelement to which the connection is being made.

TOL | Same meaning as on the SEBULK. (Values provided on SECONCT entries override the values provided on the SEBULK.)

LOC | Same meaning as on SEBULK. (Values provided on SECONCT entries override the values provided on the SEBULK.)

GIDAi, GIDBi | Pairs of grid points to be connected. GIDAi in SEIDA are connected to GIDBi in SEIDB if the coincident tolerance test is satisfied (unless overridden using LOC).

Example:

```
<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
</tr>
</thead>
<tbody>
<tr>
<td>SECONCT</td>
<td>10</td>
<td>20</td>
<td>1.0E-4</td>
<td>YES</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
<pre><code>   | 1001 | 4001 | 1002 | 4002 | 2222 | 4444 |
</code></pre>
```

This entry states that when connecting PART 10 to superelement 20, the tolerance for the coincident grid point test will be 1.0E-4 units and the coincident point test will be performed. This entry also states that we wish to connect point 1001 in PART 10 to point 4001 in superelement 20, point 1002 in PART 10 to point 4002 in superelement 20, and point 2222 in PART 10 to point 4444 in superelement 20. (In this context, point can apply to sets of either GRID entries or SPOINTs.)

**SEEXCLD**

The format of the SEEXCLD entry follows:

**Format:**

```
<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
</tr>
</thead>
<tbody>
<tr>
<td>SEEXCLD</td>
<td>SEIDA</td>
<td>SEIDB</td>
<td>GIDA1</td>
<td>GIDA2</td>
<td>GIDA3</td>
<td>GIDA4</td>
<td>GIDA5</td>
<td>GIDA6</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
<pre><code>   | GIDA7 | GIDA8 | -etc.- |   |   |   |   |    |
</code></pre>
```
This entry is used to limit the automatic search logic. While the SEBNDRY limits the search to selected grid points, the SEEXCLD excludes grid points from the search. Any grid points listed on this entry are grid points in SEIDA that the automatic logic cannot connect to grid points in SEIDB.

Description of the fields on this entry:

SEIDA   PART for which this entry is used in the automatic search routines.

SEIDB   PART(s) for which the search logic uses this list when searching for points coincident to the ones in SEIDA, listed here. This field contains either an integer superelement ID or the word ALL if the list is used for all superelements, when searching for coincident points for SEIDA.

GIDAi  Grid points in SEIDA that cannot be connected to SEIDB if there are coincident points in SEIDB.

Example 2:

<table>
<thead>
<tr>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>SEEXCLD</strong></td>
<td>110</td>
<td>10</td>
<td>45</td>
<td>678</td>
<td>396</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

The above entry states that when connecting PART 110 to superelement 10, grid points 45, 678, and 396 in PART 110 are not considered by the automatic logic.

To show another example of the usage of these entries, we will look at the model used earlier in this section. When we are connecting PART 1 to PART 3, grid points 35 and 36 (which have the same number in both PARTs) are used. This connection can be made in several ways: The example used the automatic connection logic, which determines that these grid points are coincident and connects them; or we could connect these PARTs manually, using the SECONCT entry as follows:

<table>
<thead>
<tr>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>SECONCT</strong></td>
<td>1</td>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>35</td>
<td>35</td>
<td>36</td>
<td>36</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

This entry would not replace the automatic search logic, rather it would confirm that we want to make this connection, in addition to any found by the search logic. If a SEBULK entry were included with METHOD set to MANUAL, then this entry would replace the automatic search logic.

One way to simplify the internal search is to tell the program that only grid points 35 and 36 in PART 1 can be connected to any other superelement.

<table>
<thead>
<tr>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>SEBNDRY</strong></td>
<td>1</td>
<td>ALL</td>
<td>35</td>
<td>36</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
default, the automatic logic connects any coincident grid points. Using these entries, you can change from fully automatic to semi-automatic (SECONCT plus automatic or the SEBNDRY or SEEXCLD) or manual (METHOD=MANUAL on the SEULK with a SECONCT). The choice is yours. The logic for the automatic connection is very efficient. In our testing, it took only a few seconds to search over 1,000,000 grid points to find all coincident points in a model. Compared to the amount of time to be spent solving this problem, this amount of time is negligible.

Manually Defining Exterior Points for a PART

When defining a PART it is possible to use ASET, BSET, CSET and/or ASET1, BSET1, CSET1 entries to define exterior points. Any point in a PART that is listed on either an ASET or ASET1 entry is automatically made exterior to that PART (whether there is a coincident point in another superelement or not). All six DOFs of that point are made exterior. The fields on these entries that are used to select DOFs are ignored when these entries are used in PARTs. If there is no coincident point in any other PART, internal points are created in the residual structure that are coincident with the selected points, and the matrices for the PART are connected to these points.

Moving and/or Rotating PARTs

If the model of a PART is not in the correct location or orientation, you can move and/or rotate the part with NX Nastran before the program tries to connect it to the other components of the model. This moving and rotating is accomplished using the SELOC entry in the Main Bulk Data Section.

Format:

<table>
<thead>
<tr>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
</tr>
</thead>
<tbody>
<tr>
<td>SELOC</td>
<td>SEID</td>
<td>PA1</td>
<td>PA2</td>
<td>PA3</td>
<td>PB1</td>
<td>PB2</td>
<td>PB3</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

This entry instructs NX Nastran to move and/or rotate a PART using three non-collinear GRID entries in the PART and three GRID entries or POINTs (defined in the Main Bulk Data Section), which are located where you would like those entries (in the PART) to be.

The fields on this entry are described as follows:

SEID  PART that is being translated and/or rotated.

PA1-PAB Three GRID entries in the PART that are to be used to move the PART. After moving, these points will be coincident with PB1-PB3.

PB1-PB3 Three points (either GRID or POINT entries) defined in the Main Bulk Data Section that define where the PART should be.

When this entry is encountered, the program relocates the PART so that grid points PA1-PA3 in the PART are coincident with PB1-PB3 in the main bulk data. This repositioning is performed before the coincident grid point search logic is performed, thus connections for the PART are based on the relocated position. When a PART is relocated like this, the basic coordinate system for the PART is translated and rotated to move the PART into the desired position. Output for the PART is relative to the original basic system.
If you request NX Nastran plots of the model, any PART that has been translated and/or rotated will appear in its final location. Pre- or post-preprocessors don't display the PART in any location other than its original location and orientation. Therefore, we recommend that you use the NX Nastran plotter to view your model after using any SELOC entries. Also, deformed plots are available with PARTs in their final locations. There is an option (SENOMOVE) on the PLOT command that can be used to display PARTs in their original location.

This is a good time to discuss the main bulk data basic coordinate system. When you use PARTs to define a model, each PART is defined in a separate section of the bulk data. As a default, NX Nastran assumes that the basic coordinate system for all PARTs is identical to that defined in the Main Bulk Data Section. Therefore, we have a term known as the main bulk data basic coordinate system. As long as your model is defined using the main bulk data basic coordinate system, everything works correctly. But what if different engineers use different basic coordinate systems when modeling components? How do you tell NX Nastran the basic coordinate system for PART 5 is different than that for PART 3?

This differentiation is the purpose of the SELOC entry. If the basic coordinate system for any PART is different than that used in the Main Bulk Data Section, you must tell the program how to connect that PART to the rest of the model. (That is, the PART will be in the wrong location if we assume that its basic coordinate system is identical to that in the main bulk data.) It is often difficult to specify the translation and rotation needed to align a PART with the rest of the model; therefore, with the SELOC entry you can use points to define where you want the PART to be and in what orientation.

For example, let us assume that we have the following model, with PART 100 modeled in the wrong location.
Unless we either manually connect grid points 350 and 360 in PART 100 to grid points 35 and 36 in PART 3 or use the SELOC to relocate PART 100 to the correct position, PART 100 will not be connected to the rest of the model, which is incorrect.

If we manually connect the two, the answers will be correct, but any plots will look incorrect. (PART 100 will be plotted in its original position and will appear to not be connected to the rest of the model.) Using the SELOC entry, we can move PART 100 into the correct location. First we need three non-collinear grid points in PART 100. We select points 350, 360, and 980. (Any three non-collinear points would work, but because we know we want grid 350 to connect to grid 35 in PART 3 and grid 360 to connect to grid 36 in PART 3, we use these GRIDs as two of the points on the SELOC.)

There are no points in the main bulk data that are coincident with the locations where we want to place these points in PART 100, thus we must add them. There is an entry in the main bulk data that can be used for this purpose, called a POINT. The format of this entry is identical to that of a GRID, with the exception that it has only geometry. There is no CD, PS, or SEID field on this entry, because the POINT simply defines a location in space, not DOFs.
Format:

<table>
<thead>
<tr>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
</tr>
</thead>
<tbody>
<tr>
<td>POINT</td>
<td>ID</td>
<td>CP</td>
<td>X1</td>
<td>X2</td>
<td>X3</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

The advantage is obvious: Because a POINT has no DOFs, the automatic connection logic will not connect a GRID from a PART to a POINT. The logic can only connect a GRID to a GRID.

Using this, we define three points in the main bulk data that will be used to locate PART 100. We use POINT numbers 10001, 10002, and 10003. These points are well outside the range of grid point IDs used in the model, so there is little chance for confusion.

The input for this example would look as follows (selected entries shown):

```plaintext
SOL 101 $ or whatever solution you are using
CEND
TITLE = show re-locating a PART
$ $ rest of Case Control not shown $ $ BEGIN BULK $ $ MAIN BULK DATA - only POINTs and SELOC shown for moving PART 100 $ the model is not shown here $ $ POINT,10001,-3.6,6.,0.
POINT,10002,-2.8,6.,0.
POINT,10003,-1.2,10.,0.
SELOC,100,350,360,980,10001,10002,10003 $ $ end of MAIN BULK DATA SECTION $ $ BEGIN SUPER=100 $ $ input for PART 100 not shown, except GRID points 350, 360, and 970 $ $ GRID,350,-5.2,8.,0.
GRID,360,-4.4,8.,0.
GRID,980,-2.8,12.,0.
$ $ BEGIN SUPER=3 $ $ grid points 35, 36, and 98 shown for reference $ $ GRID,35,-3.6,6.,0.
GRID,36,-2.8,6.,0.
GRID,98,-1.2,10.,0.
$ $ rest of the input file $ $ . . ENDDATA
```

In this file, PART 100 is offset by 1.6 units in the X-direction and 2.0 units in the Y-direction. In the Main Bulk Data Section, we define POINTs (once again, these are simply locations in space and have no associated DOFs) 10001, 10002, and 10003 at the locations where we would like to place grid points 350, 360, and 980 from PART 100. We then use the SELOC entry to move PART 100 to the desired location. Grid points 350 and 360 (in PART 100) would be connected to grid points 35 and 36 (in PART 3), respectively, and the solution would be for the entire model, assembled as follows (not all grid points or elements are shown):
If the repositioning involves rotating the PART, then the basic coordinate system of that PART will be rotated. Thus, all loadings, MPCs, and SPCs go through the same rotation as the PART—possibly causing unexpected results if you have not planned for this rotation. There are some loading entries that account for this rotation automatically. Currently, only the GRAV and RFORCE entries support this capability, accomplished by the field labeled MB on these entries. The formats for these two entries are:

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
</tr>
</thead>
<tbody>
<tr>
<td>GRAV</td>
<td>SID</td>
<td>CID</td>
<td>A</td>
<td>N1</td>
<td>N2</td>
<td>N3</td>
<td>MB</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
</tr>
</thead>
<tbody>
<tr>
<td>RFORCE</td>
<td>SID</td>
<td>G</td>
<td>CID</td>
<td>A</td>
<td>R1</td>
<td>R2</td>
<td>R3</td>
<td>METHOD</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

The MB field on these two entries tells the program whether the loading is defined using a coordinate system defined in the PART (and therefore rotated with the PART), which is the default, or whether the coordinate system is defined in the Main Bulk Data Section (MB=-1). If the load is defined using a coordinate system from the Main Bulk Data Section, the loading does not rotate with the PART; rather, the loading retains the original orientation based on the selected coordinate system in the Main Bulk Data Section.
2.2 Defining Superelements in the Main Bulk Data Section

Superelement Definition

As the name implies, main bulk data superelements are defined to NX Nastran in the Main Bulk Data Section, and the input data is partitioned into separate sections based on a list of interior grid points that you provide. The Main Bulk Data Section is defined as the information contained between the BEGIN BULK and either ENDDATA or the first BEGIN BULK SUPER= entry.

Main bulk data superelements are defined using field 9 on the GRID entry and/or by using SESET entries to provide a list of interior points.

In addition, a superelement can be defined as a copy of another superelement (image superelement) or by using matrices from an external source (external superelement).

The following illustration shows how superelements are defined using GRID or SESET entries.

```
<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>GRID</td>
<td>GID</td>
<td>CP</td>
<td>X1</td>
<td>X2</td>
<td>X3</td>
<td>CD</td>
<td>PS</td>
<td>SEID</td>
<td></td>
</tr>
</tbody>
</table>

Superelements are identified by an integer
```

or

```
<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
</tr>
</thead>
<tbody>
<tr>
<td>SESET</td>
<td>SEID</td>
<td>G1</td>
<td>&quot;THRU&quot;</td>
<td>G2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

```

where G1, G2, G3... are grid or scalar identification numbers.

Both GRID and SESET entries can be used in the same run. If there is a conflict between field 9 on a GRID entry and an SESET entry, the SESET takes precedence. For example, if the following two bulk data entries appeared in the same input file, grid point 47 would belong to the residual structure (SEID=0), because that membership is defined on the SESET entry, which would override the specification (SEID=2) on the GRID entry:

```
<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>GRID</td>
<td>47</td>
<td>0.</td>
<td>100.</td>
<td>50.</td>
<td>7</td>
<td>8</td>
<td>9</td>
<td>10</td>
<td></td>
</tr>
<tr>
<td>SESET</td>
<td>0</td>
<td>1</td>
<td>THRU</td>
<td>50</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

```

There is no limit on the number of SESET entries that can be used to define a superelement, and with the THRU option on the SESET entry, you can have open sets. That is, not all grid points in the range specified need to exist. If a nonexistent grid point is referenced by an SESET entry, that part of the entry is ignored.

Any grid points defined in the main bulk data that are not assigned to a superelement, using either a SESET or GRID entry, automatically belong to the residual structure (SEID 0). If SPOINTs or
EPOINTs exist in the input stream, they automatically belong to the residual structure and can not be interior to a superelement.

**Interior Versus Exterior Points**

Any grid point that you assign to a superelement is interior to that superelement. Exterior grid points are defined by the program based on processing order. If a grid point is connected to a superelement but is interior to a downstream superelement (one which is processed later), that grid point is exterior to the upstream superelement. Processing order can be defined by you (see “Multilevel Superelement Analysis”) or the program. For now, we will limit the discussion to what is known as a single-level processing tree, where all exterior points of the superelements are interior to the residual structure.

**Bulk Data Partitioning**

The Main Bulk Data Section is partitioned based on user input. (This is done internally using tables, but it is easier to visualize in terms of the bulk data.) Once the program detects either SESET entries or entries in field 9 of a GRID entry, the input is partitioned. A separate set of bulk data is created for each superelement, based on input definitions you provide.

The partitioning process for the main bulk data is as follows:

1. The processing order is defined for the model.

2. Bulk data is partitioned for each superelement in the same order as the processing occurs. All grid points (interior) assigned to the superelement by SESET and GRID entries are removed from the main bulk data and placed in a separate area for the superelement.

3. All elements (interior) connected to the interior grid points of the superelement are removed.

4. Copies are made of all exterior points (grid points that are not interior, but are connected by the elements contained within the superelement).

   Any loading entries specific to the interior grid points and/or interior elements are removed from the bulk data. Any information removed from the bulk data for a superelement is not available for use in any other superelement. Copies are made of common data-for example, property entries and GRAY entries. Common information is not removed, rather this information is copied so that it will be available for the other superelements.

5. Once all information for the first superelement has been separated, the process is repeated for the next superelement. After the bulk data has been separated for all superelements, any remaining data belongs to the residual structure.

Once the main bulk data has been partitioned, each superelement is processed using its own unique bulk data. If there are entries that need information from more than one superelement for processing (for example, PLOAD4 entries with a THRU range including elements from several superelements), the program may give a fatal message when processing the affected superelements. To simplify matters, think of each superelement as being processed independently from all others, thus reducing the chance for errors.

**Grid Point Partitioning**
Interior grid points are assigned by the user. Exterior points are normally selected by the program, based on element connectivity. Any grid point (that is not interior to the superelement) connected to an element that is interior to a superelement is defined as an exterior point for the superelement.

In addition, you can define additional exterior grid points beyond the ones assigned by the program. You can do this accomplished using the CSUPEXT entry.

<table>
<thead>
<tr>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
</tr>
</thead>
<tbody>
<tr>
<td>CSUPEXT</td>
<td>SEID</td>
<td>GID1</td>
<td>GID2</td>
<td>GID3</td>
<td>-etc-</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

This entry is used to assign exterior points that might not be assigned based on connectivity. Examples are: reference grid points on CBAR and CBEAM entries, PARAM,GRDPNT, reference grid points on loading entries, and points connected only by MPC entries.

In the main bulk data, SPOINTs and EPOINTs are always interior to the residual structure.

There are special rules regarding scalar (SPOINT) and extra (EPOINT) points. In the main bulk data, both scalar and extra points are, by definition, interior to the residual structure. They can not be interior to any other superelement. Scalar points can be exterior to any number of superelements and are often used in dynamics to represent component modes.

Extra points can not be exterior to any superelement; in fact, any entries that reference extra points can reference only points and elements that are interior to the residual structure.

More information on SPOINTs and EPOINTs is presented in the section on dynamic analysis.

**Element Partitioning**

Element partitioning in the Main Bulk Data Section is automatically handled by NX Nastran. Any element that is connected only to the interior points of a superelement is assigned to that superelement. A branch element is an element that connects to grid points belonging to more than one superelement. Branch elements are assigned to the most upstream (based on processing order) superelement to which they are connected. A boundary element is an element that is connected to grid points, all of which are exterior to a superelement. Boundary elements are automatically sent downstream, but may be placed in the upstream superelement by using the SEELT Bulk Data entry.

<table>
<thead>
<tr>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
</tr>
</thead>
<tbody>
<tr>
<td>SEELT</td>
<td>SEID</td>
<td>EID1</td>
<td>EID2</td>
<td>EID3</td>
<td>-etc-</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

or

<table>
<thead>
<tr>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
</tr>
</thead>
<tbody>
<tr>
<td>SEELT</td>
<td>SEID</td>
<td>EID1</td>
<td>“THRU” EID2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Either form of this entry can be used. Once again, multiple SEELT entries can occur for any superelement.

Concentrated mass elements (CONMi) are assigned as interior to the superelement that contains the grid point to which the element is attached.

**Example of Bulk Data Partitioning**

This example uses the sample file and model shown in “Introduction and Fundamentals”.
Figure 2-2 shows the model divided into superelements. The grid points shown are the grid points that belong to the residual structure. All other grid points belong to one of the superelements. These residual structure points are all exterior points for one or more of the superelements. The residual structure contains not only the reduced matrices for the superelements, but also any elements that were not assigned to another superelement. The residual structure for this model contains only one element, QUAD4 number 5, which was not assigned to another superelement by the program. As shown, the model is what we have described as a single-level tree as shown below in Figure 2-6.

In this type of model, all points that are exterior to any superelements are interior to the residual structure. In this way, all superelements attach directly to the residual structure. If the exterior point of any superelement is an interior point of another superelement, the model is what is known as a multilevel tree, which will be discussed later ("Multilevel Superelement Analysis").
Now we will look at how the data is partitioned for superelement 1. The first thing NX Nastran does after sorting the bulk data is to partition the bulk data, based on the user-supplied SESET and GRID definitions. This action results in the program placing grid points 33, 34, 37, 38, 45 through 50, 57 through 62, 69 through 74, 81 through 86, and 93 through 98 into a separate area for superelement 1. These points are the interior grid points for superelement 1. The next step is to remove all elements connected to these grid points from the bulk data and add them to the set for superelement 1. These elements are the interior elements (elements 18 through 42) for superelement 1. The program next finds grid points that are connected to these elements but are not interior to superelement 1 and copies them into the set for superelement 1. These grid points (35 and 36) are the exterior grid points for superelement 1. Copies of the PSHELL and the MAT1 entries are also made. There is now a unique set of bulk data that describes superelement 1, and the superelement can be processed independently from the rest of the model. Because the input as shown has no SPC, MPC, or loading entries, the partitioning of those entries will not be shown at this time. If such entries existed, entries unique to the superelement would be partitioned out and placed in the set for superelement 1, and information that was shared would be copied into the information for superelement 1 (see “Loads, Constraints, and Case Control in Static Analysis” for a description of this process).
The partitioned model for superelement 1 is shown in Figure 2-4. The exterior points are marked with x. During the processing, NX Nastran replaces this finite element model of superelement 1 with a set of reduced matrices that represent the mass, stiffness, and damping properties of superelement 1, as seen by the adjoining structure. In later chapters, this process is shown in a manner similar to that shown in Figure 2-5.

Elements 18 through 42 are interior to superelement 1. Elements 19, 20, and 21 are known as branch elements, because they are connected to grid points belonging to both superelement 1 and the residual structure. As mentioned before, these elements are assigned to the upstream superelement, or superelement 1.

If a CBAR element was connected to grid points 35 and 36, it would be a boundary element (i.e., connected to grid points that were both exterior to superelement 1). By default, the CBAR would belong to the residual structure. However, the CBAR could be moved into superelement 1 by using an SEELT entry. For example, if the following CBAR element connectivity existed,

<table>
<thead>
<tr>
<th></th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
</tr>
</thead>
<tbody>
<tr>
<td>CBAR</td>
<td>1000</td>
<td>1</td>
<td>35</td>
<td>36</td>
<td>71</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

then a FATAL error would occur (since GRID 71 is used for orientation) unless the CBAR was assigned to superelement 1, using the following SEELT entry.

<table>
<thead>
<tr>
<th></th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
</tr>
</thead>
<tbody>
<tr>
<td>SEELT</td>
<td>1</td>
<td>1000</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

The reason for this error is as follows: NX Nastran looks only at the element connectivity when assigning elements to superelements. CBAR 1000 connects to grid points 35 and 36 and finds they both belong to the residual structure. The CBAR is then assigned interior to the residual structure. Because of this, without the SEELT entry, a FATAL message occurs when processing the residual structure. When element 1000 is processed, the program searches the bulk data of the residual structure for grid point 71, but cannot find it, because it is a part of superelement 1. The way to prevent this problem is to include the SEELT entry; then BAR 1000 belongs to superelement 1, which has all the data necessary to process it without any problems. You could also use GID 19 (which is in the residual structure) instead of 71.
In Figure 2-5 the exterior points are shown; the superelement is represented by a circle containing the SEID, which is connected to the exterior points.

The Superelement MAP - SEMAP

The SEMAP is a roadmap for superelement solutions, indicating superelement membership and processing order.

Whenever superelements are used in a solution, NX Nastran prints a summary of the grid point and element partitioning for the model. This printout is called the SEMAP. The SEMAP output includes a listing of interior and exterior points and elements for all superelements. In addition, it includes the processing order for the superelements and CPU and disk storage estimates for each superelement.

The SEMAP is printed by default for all runs using superelements. If you do not want the SEMAP printout, PARAM,SEMAPPRT,-1 should be added to the bulk data.

Contents of the Superelement Map

Superelement Definition Table

The first item in the output for the SEMAP is Table 2-1 Superelement Definition Table. The contents of this table include the processing order for the superelements that have been defined and the superelement tree. For the sample problem the superelement definition table appears as follows:

<table>
<thead>
<tr>
<th>INDEX</th>
<th>SUPERELEMENT</th>
<th>PROCESS ORDER</th>
<th>DOWNSTREAM SE</th>
<th>SUPERELEMENT TYPE</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>8</td>
<td>0</td>
<td>RESIDUAL STRUCTURE</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>PRIMARY SUPERELEMENT</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>2</td>
<td>0</td>
<td>PRIMARY SUPERELEMENT</td>
</tr>
<tr>
<td>3</td>
<td>3</td>
<td>3</td>
<td>0</td>
<td>PRIMARY SUPERELEMENT</td>
</tr>
<tr>
<td>4</td>
<td>4</td>
<td>4</td>
<td>0</td>
<td>PRIMARY SUPERELEMENT</td>
</tr>
<tr>
<td>5</td>
<td>5</td>
<td>5</td>
<td>0</td>
<td>PRIMARY SUPERELEMENT</td>
</tr>
<tr>
<td>6</td>
<td>6</td>
<td>6</td>
<td>0</td>
<td>PRIMARY SUPERELEMENT</td>
</tr>
<tr>
<td>7</td>
<td>7</td>
<td>7</td>
<td>0</td>
<td>PRIMARY SUPERELEMENT</td>
</tr>
</tbody>
</table>

Table 2-1. Superelement Definition Table for the Sample Problem
The superelement definition table is printed twice: first, in ascending order of superelements, and second, in the order in which the superelements are processed by NX Nastran. The table also identifies the downstream superelement (once again, this is used mainly for multilevel analysis—see “Multilevel Superelement Analysis”). In the case of this model, which is a single-level model, all superelements have the residual structure (SEID=0) as a downstream superelement.

The farthest right-hand column has a description of the type of each superelement. In this example, all superelements are primary superelements, meaning that each superelement has its own unique bulk data. The other types of superelements are: identical and mirror images, and external superelements.

The next set of information is the table of downstream connections for each superelement. This table is useful in identifying multilevel configurations. For a single-level tree, as shown, the table only verifies what we already know.

The next part is the superelement tree. In printed form, a representation of the processing order resembles a tree. For a single-level tree, the processing order only repeats what we already know; however, for a multilevel tree, the order is important not only for initial processing, but for restarts, in that it shows the path to follow when changes are made. Think of the processing order as a tree, where the processing must flow down the tree and data recovery is performed by climbing the tree. Figure 2-6 helps explain the printout and shows a graphical representation of the tree.

The printed form shows the processing tree sideways. In the printed form, each superelement appears as its SEID. Below each superelement ID appears the processing order for that

<table>
<thead>
<tr>
<th>Table 2-1. Superelement Definition Table for the Sample Problem</th>
</tr>
</thead>
<tbody>
<tr>
<td>INDEX</td>
</tr>
<tr>
<td>-------</td>
</tr>
<tr>
<td>1</td>
</tr>
<tr>
<td>2</td>
</tr>
<tr>
<td>3</td>
</tr>
<tr>
<td>4</td>
</tr>
<tr>
<td>5</td>
</tr>
<tr>
<td>6</td>
</tr>
<tr>
<td>7</td>
</tr>
</tbody>
</table>

The SUPERELEMENT TREE

( NO. LEVELS = 1  NO. TIPS = 7  *= PROCESS ORDER )

<table>
<thead>
<tr>
<th>TIP INDEX</th>
</tr>
</thead>
<tbody>
<tr>
<td>-1-</td>
</tr>
<tr>
<td>1</td>
</tr>
<tr>
<td>1*</td>
</tr>
<tr>
<td>2</td>
</tr>
<tr>
<td>2*</td>
</tr>
<tr>
<td>3</td>
</tr>
<tr>
<td>3*</td>
</tr>
<tr>
<td>4</td>
</tr>
<tr>
<td>4*</td>
</tr>
<tr>
<td>5</td>
</tr>
<tr>
<td>5*</td>
</tr>
<tr>
<td>6</td>
</tr>
<tr>
<td>6*</td>
</tr>
<tr>
<td>7</td>
</tr>
<tr>
<td>7*</td>
</tr>
</tbody>
</table>
superelement. Once again, because this is a simple problem, the processing order matches the SEID, which is generally not the case in more complicated problems.

![Superelement Tree](image)

**Figure 2-6. Graphical Superelement Tree.**

*Figure 2-6 shows the superelement tree in a more convenient form. In this form, we can see that each superelement has its own branch and, therefore, its own path. Each superelement attaches only to the residual structure (SEID=0) and can be processed independently from all other superelements. When a model is defined in this manner, it is known as a single-level tree. For processing, the order goes down the tree, or directly from each superelement to the residual, indicating that all superelements must be processed before the residual structure (assembly) can be solved. Once the problem is solved, data recovery progresses up the tree, meaning that once the residual structure solution is available, data recovery can be performed for any of the superelements in this model in any order. If we had a multilevel tree, there would be paths that must be followed for obtaining desired superelement results (see “Multilevel Superelement Analysis”).

### 2.3 Defining External Superelements

An external superelement is a reduced boundary representation of a full component model stored in an external file. The reduced boundary representation for an external superelement includes:

- Boundary matrices like stiffness, mass, and so on written in terms of the boundary point degrees-of-freedom.
- Geometric coordinates of the boundary points.
- Output transformation matrices (OTMs).

After you create one or more external superelements during component solutions, you can then combine them with the residual structure in a system solution.

### External Superelement Procedure Overview

Prior to NX Nastran 6, the external superelement procedure required three steps:

1. Create external superelements during component solutions. One component solution is required for each external superelement.

2. Assemble the external superelements to the residual structure and perform a system solution.
3. Recover results for each external superelement during restart runs of the component solutions using the results of the system solution. One restart run is needed for each external superelement.

This procedure is still supported and is documented in the NX Nastran Quick Reference Guide under the EXTOUT parameter.

A more efficient procedure became available in NX Nastran 6. In the new procedure, results recovery for external superelements is integrated into the system solution. Now you can recover results for all external superelements from a single solution and optionally combine the results for all the external superelements and the residual structure into a single data block. By combining all the results into a single data block, you can examine the external superelement results in the context of the entire system.

The new external superelement procedure can be summarized in two steps as follows:

1. Create external superelements from component solutions. One component solution is still required for each external superelement.

2. Assemble the external superelements and residual structure into a single model, solve the model, and recover results for the external superelements and residual structure from a system solution.

Component Solutions

One component solution is required for each external superelement you want to create. During each component solution, NX Nastran stores the reduced boundary representation of the superelement externally. You can optionally specify the storage media for the external file from either database (.dball), .op2, .op4 (formatted or unformatted), or .pch. You can also optionally direct NX Nastran to store connectivity data for the external superelement on an .asm file. The connectivity data defines how the external superelement is attached to the residual structure and other external superelements. If you do not direct NX Nastran to store the connectivity data, you have to manually create a file containing this data.

System Solution

For the system solution, NX Nastran requires:

- The reduced boundary representation for each external superelement.
- The connectivity data for each external superelement.
- The bulk data defining the residual structure.

NX Nastran automatically retrieves the reduced boundary representation for each external superelement from their respective storage media. However, to retrieve the connectivity data, you must direct NX Nastran to the files containing this data for each external superelement. You must also provide an input file for the system solution that contains the bulk data for the residual structure and whether or not to combine all of the results into a single data block.

Specific NX Nastran input file requirements for both component solutions and system solutions are described in the following sections for the new procedure.
Figure 2-7. External Superelement Procedure

**NX Nastran Input File Requirements for Component Solutions**

When using the new procedure, the NX Nastran input file for an external superelement component solution must include:

- An EXTSEOUT case control command to create the external superelement. When using EXTSEOUT, NX Nastran generates the reduced boundary representation and output transformation matrices (OTMs). Depending on the describers you specify with EXTSEOUT, you can control which boundary matrices are stored, the media on which the boundary matrices are stored, and whether to write connectivity data to an .asm file.

```
EXTSEOUT(ASMBULK,EXTBULK,EXTID=100,MATOP4=30)
```

- An ASET bulk entry (or, alternately, combinations of BSET and CSET bulk entries) to designate which grids represent boundary points. Boundary points are grids on the external superelement where constraints or loads are applied or where the external superelement connects to either the residual structure or other external superelements.

```
ASET1,123456,104,108
```

For a dynamic analysis that uses the component mode synthesis method, you can optionally control whether or not all the q-set DOF are retained in the definition of the external superelement. To do so, use the QSETREM parameter.

- Specify QSETREM = YES (default) to remove unused q-set DOF from the definition of the external superelement. When QSETREM = YES and the model contains mechanisms, mechanism modes are calculated, but not retained.

- Specify QSETREM = NO to retain all q-set DOF in the definition of the external superelement. When QSETREM = NO and the model contains mechanisms, mechanism modes are calculated and retained.
Note

Specifying QSETREM = NO can lead to slow processing times for stresses, strains, and forces, especially if a large number of q-set DOF are defined, but not associated with calculated modes.

By specifying various describers in the EXTSEOUT case control command, you can control:

• The boundary matrices that are stored. By default, all the boundary matrices are stored. However, you can specify that a subset of the boundary matrices be stored by specifying a subset of the STIFFNESS, MASS, DAMPING, K4DAMP, and LOADS describers.

• The media on which the boundary matrices are stored. The boundary matrices can be stored on either database (.dball), .op2, .op4 (formatted or unformatted), or .pch files.

If none of either the MATDB (or MATRIXDB), DMIGDB, DMIGOP2, DMIGPCH, MATOP4 (or MATRIXOP4) describers is specified, the boundary matrices are stored on the database by default.

If the EXTBULK describer is specified along with either the MATDB (or MATRIXDB), DMIGDB, DMIGOP2, DMIGPCH, or MATOP4 (or MATRIXOP4) describer, the boundary matrices are stored as follows:

<table>
<thead>
<tr>
<th>Descriptor specified</th>
<th>Storage media for boundary matrices</th>
</tr>
</thead>
<tbody>
<tr>
<td>MATDB (or MATRIXDB)</td>
<td>.pch</td>
</tr>
<tr>
<td>DMIGDB</td>
<td>.pch</td>
</tr>
<tr>
<td>DMIGOP2</td>
<td>.pch</td>
</tr>
<tr>
<td>DMIGPCH</td>
<td>.pch</td>
</tr>
<tr>
<td>MATOP4 (or MATRIXOP4)</td>
<td>.op4</td>
</tr>
</tbody>
</table>

If the EXTBULK describer is not specified and one of either the MATDB (or MATRIXDB), DMIGDB, DMIGOP2, DMIGPCH, MATOP4 (or MATRIXOP4) describers is specified, the boundary matrices are stored as follows:

<table>
<thead>
<tr>
<th>Descriptor specified</th>
<th>Storage media for boundary matrices</th>
</tr>
</thead>
<tbody>
<tr>
<td>MATDB (or MATRIXDB)</td>
<td>.dball</td>
</tr>
<tr>
<td>DMIGDB</td>
<td>.dball</td>
</tr>
<tr>
<td>DMIGOP2</td>
<td>.op2</td>
</tr>
<tr>
<td>DMIGPCH</td>
<td>.pch</td>
</tr>
<tr>
<td>MATOP4 (or MATRIXOP4)</td>
<td>.op4</td>
</tr>
</tbody>
</table>

Note

Beginning with NX Nastran 7.1, during the system solution, external superelement data is automatically retrieved from the media where the boundary matrices are stored. As such, the EXTBULK describer is only needed if you want the external superelement data to be written to a .pch file.

• The method of assembling the external superelement into the system model.
If the ASMBULK descriptor is specified, SEBULK, SECONCT, GRID, and CORD2x bulk entries are automatically generated and stored on an .asm file. These bulk entries represent assembly and boundary point data. The data on the .asm file is later included with the bulk data for the residual structure during the system solution.

If the ASMBULK descriptor is not specified, you must manually add SEBULK, SECONCT, GRID, and CORD2x bulk entries to the bulk section of the system solution input file.

- The GEOM descriptor can be used to request that the software include the full finite element geometry with the external superelement output. The full finite element geometry output can be used by pre/post processors for data recovery and post processing on the full FE model. The request writes the geometry data blocks GEOM1EXA, GEOM2EXA, and GEOM4EXA with the external superelement. The descriptor is supported for the MATDB (or MATRIXDB), DMIGDB, and DMIGOP2 storage options.

- The LOAD descriptor can be used to store static and dynamic loads with an external superelement. See External Superelement Loads.

To include differential stiffness in the definition of an external superelement, two subcases are required in the creation run. The first subcase is a static subcase that references the stress-stiffening loads. Under the first subcase, include the second subcase. In the second subcase, include STATSUB = n, where n is the identification number of the static subcase. The software will then calculate the modal content of the external superelement from a stiffness formulation that includes stiffness contributions resulting from static stiffening. Always include the EXTSEOUT case control command above the subcase level. An example of the required setup is as follows:

```
...  
SOL 112
CEND
TITLE = ...
EXTSEOUT(...) 
$SUBCASE 1
   $ STATIC SUBCASE
   LOAD = 11
$SUBCASE 2
   $ DYNAMIC SUBCASE
   TSTEP = 100
   STATSUB = 1
   METHOD = 10
   DLOAD = 20
BEGIN BULK
...  
```

For accuracy and consistency, the loads used to generate differential stiffness for the external superelement during the creation run should be the same loads used in the system run without any scaling. If the loads are scaled by a non-unity scaling factor from a case control command like P2G or a bulk entry like LOAD, the differential stiffness portion of the external superelement stiffness matrix will no longer be consistent with the applied loads.

The ability to generate an external superelement including differential stiffness effects is available for SOLs 103, 107-112, and 187. For SOL 112, the IC(STATSUB,DIFFK) or IC(TZERO,DIFFK) case control commands can be used to generate differential stiffness effects instead of the STATSUB case control command.
NX Nastran Input File Requirements for System Solutions

When using the new procedure, the NX Nastran input file for an external superelement system solution must include:

- Output requests that are a subset of the output requests common to all component solutions.
  
  ```
  DISPLACEMENT=ALL
  STRESS=ALL
  ```

- A subcase for each external superelement that you want to perform results recovery for during the system solution. Also include a subcase for the residual structure if you want to perform results recovery on it during the system solution. The subcase for the residual structure must follow those for the external superelements.
  
  ```
  SUBCASE 1
  SUPER=100
  SUBCASE 2
  SUPER=200
  SUBCASE 3
  SUPER=0
  $ SUPER=0 always designates the residual structure
  ```

- The parameter SECOMB if you want the results for both internal and external superelements to be combined with results from the rest of the system model into a single data block.
  
  ```
  param,secomb,yes
  ```

The following requirements for the system solution input file depend on which describers are specified with the EXTSEOUT case control command in the component solution input file:

- If the ASMBULK describer was specified in the component creation run, you can use the INCLUDE bulk entry to add the .asm file which contains the connectivity data for the external superelement to the bulk section.
  
  ```
  INCLUDE 'superelement100.asm'
  ```

- If the ASMBULK describer was not specified in the component creation run, you must manually add SEBULK and SECONCT entries. The SELOC entry can optionally be used to relocate the superelement in the system model.
  
  ```
  ASSIGN INPUTT2='OUTDIR:extse04c_0.op2' UNIT=32
  ID,NASTRAN,extse04s
  SOL 112
  CEND
  ...
  BEGIN BULK
  $...
  $* SUPERELEMENT MODEL
  $*
  $* NX Superelement ID: 4
  SEBULK 4 EXTOP2 MANUAL 31
  SELOC 4 1 2 3 727 728 729
  SECONCT 4 0 NO +
  + 1 727 2 728 3 729 4 730
  $* NX Superelement ID: 5
  SEBULK 5 EXTOP2 MANUAL 32
  SELOC 5 1 2 3 731 732 733
  SECONCT 5 0 NO +
  + 1 731 2 732 3 733 4 734
  $*
  ```
• If the describers cause the reduced boundary representation to be stored on a .pch file, an INCLUDE bulk entry is needed to direct the software to the .pch file.

    INCLUDE 'superelement100.pch'

• If the LOAD descriptor was used in the external superelement creation run, you can optionally select and scale static and dynamic loads included with the external superelement in the system run. See External Superelement Loads.

External Superelement Loads

An optional method is available for storing loads with an external superelement. When you include an external superelement created using the method in a system run, you can optionally select and scale the individual loads.

In the external superelement creation run, the LOAD descriptor is available on the EXTSEOUT case control command. Use this descriptor to request the load storage method. When the LOAD descriptor is included, NX Nastran:

• Writes the load matrix \([P_a]\) into the output format you defined on the EXTSEOUT command.

• Organizes the load matrix \([P_a]\) such that each column corresponds to a unique load ID.

• Writes the DTI,SELOAD bulk entry into the output format you defined on the EXTSEOUT command. Each row in the DTI,SELOAD entry corresponds to a unique load ID. This entry maps the original load ID’s to the loads in the \([P_a]\) matrix.

When you include an external superelement created using this method in a system solution, you can include the SELOAD bulk entry in the residual structure portion of your system analysis input file to select the external superelement loads. The input format for the SELOAD entry is:

<table>
<thead>
<tr>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
</tr>
</thead>
<tbody>
<tr>
<td>SELOAD</td>
<td>LIDS0</td>
<td>SEID</td>
<td>LIDSE</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

where LIDS0 is the load ID used in the residual structure, SEID is the external superelement ID, and LIDSE is the original load ID stored in the \([P_a]\) matrix. LIDS0 must be referenced by a loading request to be active. For example, LIDS0 can match the value of \(n\) on a LOAD case control command, or match the EXCITEID on a RLOADi bulk entry.

Creating and selecting the load matrix \([P_a]\)

The load information stored in the load matrix and on the DTI,SELOAD bulk entry depends on the solution type and method in which loads are defined and referenced in the superelement creation run.

The following conditions apply to static solutions. For example, SOL 101:

• In a creation run, the LOAD = \(n\) case control command creates a single load in the resulting \([P_a]\) matrix. The \([P_a]\) matrix will have a single column that contains the load values referenced by \(n\). The corresponding LIDSE and EXCSE values on the DTI,SELOAD bulk entry will be the value \(n\).

To select this load in a system run, the LIDSE field on the SELOAD bulk entry should equal the value of \(n\) from the creation run.

• In a creation run, the LOADSET = \(n\) case control command is generally used to create multiple loads. Each definition of an LSEQ bulk entry will create a column in the \([P_a]\) matrix regardless of
whether or not the LSEQ is referenced by the LOADSET = n case control command. However, an un referenced LSEQ will generate a null column in the \( [P_a] \) matrix. The corresponding LIDSE and EXCSE on an DTI, SELOAD bulk entry will be the value of LID and EXCITEID, respectively, on an LSEQ bulk entry.

To select these loads in a system run, the LIDSE field on the SELOAD bulk entry should equal the value of an EXCITEID from the creation run.

- If both LOADSET and LOAD appear in the case control, LOADSET takes precedence.

The following conditions apply to dynamic solutions. For example, SOL 103:

- In a creation run, a column in the \( [P_a] \) matrix is created for each load (not enforced motion) defined on RLOADi and TLOADi bulk entries whether or not they are referenced in the case control. The corresponding LIDSE and EXCSE on the DTI,SELOAD bulk entries will both be the value of EXCITEID on the RLOADi or TLOADi bulk entry.

To select these loads in a system run, the LIDSE field on the SELOAD entry should equal the value of an EXCITEID from the creation run.

- In a creation run, the use of LOADSET or LSEQ in a dynamic solution is treated the same as in a static solution except an LSEQ bulk entry must be referenced by a LOADSET case control command in order to generate a column in the \( [P_a] \) matrix.

To select these loads in a system run, the LIDSE field on the SELOAD entry should equal the value of an EXCITEID from the creation run.

**Limitations**

- The method does not support SPCD enforced motion loads or thermal loads.

- The DMIGPCH output option on the EXTSEOUT command does not support the method. A load matrix \( [P_a] \) is written when the DMIGPCH output option is used, but only for static loads and without any unique load identifiers. The resulting static load matrix can be selected in a static system run by setting the P2G case control command equal to “PA”. For example, 

\[
P2G = PA
\]

**External Superelement Examples**

Example 1, using the default storage, MATDB (or MATRIXDB):

<table>
<thead>
<tr>
<th>Step 1: Superelement creation</th>
<th>Step 2: System run</th>
</tr>
</thead>
</table>
assign se100=see103hna.MASTER'
dblocate datablk=(extdb) convert(seid=100),
  logical=se100
$
SOL 103
TIME 5
DIAG 5,6,8,13,15
CEND
TITLE = SYSTEM RUN EXAMPLE1
SPC = 5
disp=all
stre=all
stra=all
forc=all
SUBCASE 1
  SUPER= 100
SUBCASE 2
  SUPER= 0
  METHOD= 200
BEGIN BULK
  eigrl, 200, , , 6
  include 'see103hna.asm'
  GRID, 10,, 0.0, 0.0, 0.0
  GRID, 11,, 1.0, 0.0, 0.0
  GRID, 12,, 2.0, 0.0, 0.0
  CBAR, 11, 1, 10, 11, 0.0, 1.0, 0.0
  CBAR, 12, 1, 11, 12, 0.0, 1.0, 0.0
  CBAR, 13, 1, 12, 13, 0.0, 1.0, 0.0
  CBAR, 14, 1, 13, 14, 0.0, 1.0, 0.0
  PBAR, 1, 1, 1.0,, 0.0833, 0.0833, 0.0833
  MAT1, 1, 3.0E7,, ,.3, 0.001
  FORCE, 104,105,, 10.0, 0.0, 0.0,-1.0
  SPC1, 105,123456,110
$
ENDDATA
Example 2, using DMIGDB storage option:

<table>
<thead>
<tr>
<th>Step 1: Superelement creation</th>
<th>Step 2: System run</th>
</tr>
</thead>
<tbody>
<tr>
<td>SOL 103</td>
<td>assign se100=see103hnd.MASTER'</td>
</tr>
<tr>
<td>TIME 5</td>
<td>dblocate datablk=(extdb) convert(seid=100),</td>
</tr>
<tr>
<td>DIAG 5, 6, 8, 13, 15</td>
<td>logical=se100</td>
</tr>
<tr>
<td>CEND</td>
<td>SOL 103</td>
</tr>
<tr>
<td><strong>extseout</strong>(extid=100, asmbulk, dmigdb)</td>
<td>TIME 5</td>
</tr>
<tr>
<td>TITLE = SE CREATE EXAMPLE2</td>
<td>DIAG 5, 6, 8, 13, 15</td>
</tr>
<tr>
<td>disp=all</td>
<td>CEND</td>
</tr>
<tr>
<td>stre=all</td>
<td>TITLE = SYSTEM RUN EXAMPLE2</td>
</tr>
<tr>
<td>stra=all</td>
<td>SPC = 5</td>
</tr>
<tr>
<td>force=all</td>
<td>disp=all</td>
</tr>
<tr>
<td>SPC = 105</td>
<td>stre=all</td>
</tr>
<tr>
<td>method= 100</td>
<td>stra=all</td>
</tr>
<tr>
<td>BEGIN BULK</td>
<td>force=all</td>
</tr>
<tr>
<td>aset1, 123456, 104, 108</td>
<td>SUBCASE 1</td>
</tr>
<tr>
<td>eigrl, 100, , , 8</td>
<td>SUPER= 100</td>
</tr>
<tr>
<td>spoint, 1001, thru, 1008</td>
<td>SUBCASE 2</td>
</tr>
<tr>
<td>qset1, 0, 1001, thru, 1008</td>
<td>METHOD= 200</td>
</tr>
<tr>
<td>GRID, 104, , 4.0, 0.0, 0.0</td>
<td>BEGIN BULK</td>
</tr>
<tr>
<td>GRID, 105, , 5.0, 0.0, 0.0</td>
<td>eigrl, 200, , , 7</td>
</tr>
<tr>
<td>GRID, 106, , 6.0, 0.0, 0.0</td>
<td>include 'see103hnd.asm'</td>
</tr>
<tr>
<td>GRID, 107, , 7.0, 0.0, 0.0</td>
<td>GRID, 10, , 0.0, 0.0, 0.0</td>
</tr>
<tr>
<td>GRID, 108, , 8.0, 0.0, 0.0</td>
<td>GRID, 11, , 1.0, 0.0, 0.0</td>
</tr>
<tr>
<td>GRID, 109, , 9.0, 0.0, 0.0</td>
<td>GRID, 12, , 2.0, 0.0, 0.0</td>
</tr>
<tr>
<td>GRID, 110, , 10.0, 0.0, 0.0</td>
<td>GRID, 13, , 3.0, 0.0, 0.0</td>
</tr>
<tr>
<td>CBAR, 105, 101, 104, 105, , 1.0, 0.0</td>
<td>grid, 15, , 8.0, 0.0, 0.0</td>
</tr>
<tr>
<td>CBAR, 106, 101, 105, 106, , 1.0, 0.0</td>
<td>CBAR, 11, 10, 11, 0.0, 0.0, 1.0, 0.0</td>
</tr>
<tr>
<td>CBAR, 107, 101, 106, 107, , 1.0, 0.0</td>
<td>CBAR, 12, 11, 12, 0.0, 0.0, 1.0, 0.0</td>
</tr>
<tr>
<td>CBAR, 108, 101, 107, 108, , 1.0, 0.0</td>
<td>CBAR, 13, 12, 13, 0.0, 0.0, 1.0, 0.0</td>
</tr>
<tr>
<td>CBAR, 109, 101, 108, 109, , 1.0, 0.0</td>
<td>CBAR, 14, 13, 14, 0.0, 0.0, 1.0, 0.0</td>
</tr>
<tr>
<td>CBAR, 110, 101, 109, 110, , 1.0, 0.0</td>
<td>PBAR, 1, 1, 1.0, 0.0, 0.0, 0.0, 0.0</td>
</tr>
<tr>
<td>PBAR, 101, 101, 1.0, 0.0, 0.0, 0.0, 0.0, 0.0</td>
<td>MAT1, 1, 1.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0</td>
</tr>
<tr>
<td>MAT1, 101, 3.0E7, , 0.001</td>
<td>ENDDATA</td>
</tr>
</tbody>
</table>
### Example 3, using DMIGOP2 storage option:

<table>
<thead>
<tr>
<th>Step 1: Superelement creation</th>
<th>Step 2: System run</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>assign output2='see103hng.op2', delete, unit=30 $</code></td>
<td><code>$ assign input2 for external SE $</code></td>
</tr>
<tr>
<td><code>$ SOL 103 TIME 5 DIAG 5,6,8,13,15 CEND</code></td>
<td><code>$ assign input2='see103hng.op2', unit= 30 $</code></td>
</tr>
<tr>
<td><code>extseout(asmbulk,extid=100,dmigop2=30)</code></td>
<td><code>$ SOL 103 TIME 5 DIAG 5,6,8,13,15 CEND</code></td>
</tr>
<tr>
<td><code>TITLE = SE CREATE EXAMPLE3 disp=all stre=all stra=all forc=all</code></td>
<td><code>TITLE = SYSTEM RUN EXAMPLE3</code></td>
</tr>
<tr>
<td><code>SPC = 105</code></td>
<td><code>SPC = 5</code></td>
</tr>
<tr>
<td><code>method= 100 BEGIN BULK</code></td>
<td><code>method= 100 BEGIN BULK</code></td>
</tr>
<tr>
<td><code>ASET1, 123456, 104, 108 eigrl, 100, , , 8</code></td>
<td><code>ASET1, 123456, 104, 108 eigrl, 100, , , 8</code></td>
</tr>
<tr>
<td><code>SPC = 105</code></td>
<td><code>SPC = 5</code></td>
</tr>
<tr>
<td><code>SUPER= 100 BEGIN BULK</code></td>
<td><code>SUPER= 0 BEGIN BULK</code></td>
</tr>
<tr>
<td><code>EIGR1, 100, , , 8</code></td>
<td><code>EIGR1, 100, , , 8</code></td>
</tr>
<tr>
<td><code>GRID,104,,4.0,0.0,0.0</code></td>
<td><code>GRID,104,,4.0,0.0,0.0</code></td>
</tr>
<tr>
<td><code>GRID,105,,5.0,0.0,0.0</code></td>
<td><code>GRID,105,,5.0,0.0,0.0</code></td>
</tr>
<tr>
<td><code>GRID,106,,6.0,0.0,0.0</code></td>
<td><code>GRID,106,,6.0,0.0,0.0</code></td>
</tr>
<tr>
<td><code>GRID,107,,7.0,0.0,0.0</code></td>
<td><code>GRID,107,,7.0,0.0,0.0</code></td>
</tr>
<tr>
<td><code>GRID,108,,8.0,0.0,0.0</code></td>
<td><code>GRID,108,,8.0,0.0,0.0</code></td>
</tr>
<tr>
<td><code>GRID,109,,9.0,0.0,0.0</code></td>
<td><code>GRID,109,,9.0,0.0,0.0</code></td>
</tr>
<tr>
<td><code>GRID,110,,10.0,0.0,0.0</code></td>
<td><code>GRID,110,,10.0,0.0,0.0</code></td>
</tr>
<tr>
<td><code>CBAR,105,101,104,105,0.0,1.0,0.0</code></td>
<td><code>CBAR,105,101,104,105,0.0,1.0,0.0</code></td>
</tr>
<tr>
<td><code>CBAR,106,101,105,106,0.0,1.0,0.0</code></td>
<td><code>CBAR,106,101,105,106,0.0,1.0,0.0</code></td>
</tr>
<tr>
<td><code>CBAR,107,101,106,107,0.0,1.0,0.0</code></td>
<td><code>CBAR,107,101,106,107,0.0,1.0,0.0</code></td>
</tr>
<tr>
<td><code>CBAR,108,101,107,108,0.0,1.0,0.0</code></td>
<td><code>CBAR,108,101,107,108,0.0,1.0,0.0</code></td>
</tr>
<tr>
<td><code>CBAR,109,101,108,109,0.0,1.0,0.0</code></td>
<td><code>CBAR,109,101,108,109,0.0,1.0,0.0</code></td>
</tr>
<tr>
<td><code>CBAR,110,101,109,110,0.0,1.0,0.0</code></td>
<td><code>CBAR,110,101,109,110,0.0,1.0,0.0</code></td>
</tr>
<tr>
<td><code>PBAR,101,101,1.0,.0833,.0833,.0833</code></td>
<td><code>PBAR,101,101,1.0,.0833,.0833,.0833</code></td>
</tr>
<tr>
<td><code>MAT1,101,3.0E7, , , 3.0,0.001</code></td>
<td><code>MAT1,101,3.0E7, , , 3.0,0.001</code></td>
</tr>
<tr>
<td><code>FORCE,104,105,,10.0,0.0,0.0,-1.0</code></td>
<td><code>FORCE,104,105,,10.0,0.0,0.0,-1.0</code></td>
</tr>
<tr>
<td><code>SPC1,105,123456,110</code></td>
<td><code>SPC1,105,123456,110</code></td>
</tr>
<tr>
<td><code>ENDDATA</code></td>
<td><code>ENDDATA</code></td>
</tr>
</tbody>
</table>
Example 4, using DMIGPCH storage option:

<table>
<thead>
<tr>
<th>Step 1: Superelement creation</th>
<th>Step 2: System run</th>
</tr>
</thead>
<tbody>
<tr>
<td>SOL 103</td>
<td>SOL 103</td>
</tr>
<tr>
<td>TIME 5</td>
<td>TIME 5</td>
</tr>
<tr>
<td>DIAG 5,6,8,13,15</td>
<td>DIAG 5,6,8,13,15</td>
</tr>
<tr>
<td>CEND</td>
<td>CEND</td>
</tr>
<tr>
<td>extsetout(asmbulk, dmigpch, extid=100)</td>
<td>extsetout(asmbulk, dmigpch, extid=100)</td>
</tr>
<tr>
<td>TITLE = SE CREATE EXAMPLE4</td>
<td>TITLE = SYSTEM RUN EXAMPLE4</td>
</tr>
<tr>
<td>disp=all</td>
<td>SPC = 5</td>
</tr>
<tr>
<td>stre=all</td>
<td>disp=all</td>
</tr>
<tr>
<td>stra=all</td>
<td>stre=all</td>
</tr>
<tr>
<td>forccall</td>
<td>stra=all</td>
</tr>
<tr>
<td></td>
<td>forccall</td>
</tr>
<tr>
<td></td>
<td>SUPER= 100</td>
</tr>
<tr>
<td></td>
<td>Method= 200</td>
</tr>
<tr>
<td></td>
<td>BEGIN BULK</td>
</tr>
<tr>
<td></td>
<td>set1, 123456, 104, 108</td>
</tr>
<tr>
<td></td>
<td>eigrl, 100, , , 8</td>
</tr>
<tr>
<td></td>
<td>spoint, 1001, thru, 1008</td>
</tr>
<tr>
<td></td>
<td>qsetl, 0, 1001, thru, 1008</td>
</tr>
<tr>
<td></td>
<td>setl, 0, 1001, thru, 1008</td>
</tr>
<tr>
<td></td>
<td>setl, 0, 1001, thru, 1008</td>
</tr>
<tr>
<td></td>
<td>asetl, 123456, 104, 108</td>
</tr>
<tr>
<td></td>
<td>eigrl, 100, , , 8</td>
</tr>
<tr>
<td></td>
<td>spoint, 1001, thru, 1008</td>
</tr>
<tr>
<td></td>
<td>qsetl, 0, 1001, thru, 1008</td>
</tr>
<tr>
<td></td>
<td>setl, 0, 1001, thru, 1008</td>
</tr>
<tr>
<td></td>
<td>setl, 0, 1001, thru, 1008</td>
</tr>
<tr>
<td></td>
<td>BEGIN BULK</td>
</tr>
<tr>
<td></td>
<td>asetl, 123456, 104, 108</td>
</tr>
<tr>
<td></td>
<td>eigrl, 100, , , 8</td>
</tr>
<tr>
<td></td>
<td>spoint, 1001, thru, 1008</td>
</tr>
<tr>
<td></td>
<td>qsetl, 0, 1001, thru, 1008</td>
</tr>
<tr>
<td></td>
<td>setl, 0, 1001, thru, 1008</td>
</tr>
<tr>
<td></td>
<td>setl, 0, 1001, thru, 1008</td>
</tr>
<tr>
<td></td>
<td>asetl, 123456, 104, 108</td>
</tr>
<tr>
<td></td>
<td>eigrl, 100, , , 8</td>
</tr>
<tr>
<td></td>
<td>spoint, 1001, thru, 1008</td>
</tr>
<tr>
<td></td>
<td>qsetl, 0, 1001, thru, 1008</td>
</tr>
<tr>
<td></td>
<td>setl, 0, 1001, thru, 1008</td>
</tr>
<tr>
<td></td>
<td>setl, 0, 1001, thru, 1008</td>
</tr>
<tr>
<td></td>
<td>begin bulk</td>
</tr>
<tr>
<td></td>
<td>Eigrl, 200, , , 6</td>
</tr>
<tr>
<td></td>
<td>include 'see103hnj.asm'</td>
</tr>
<tr>
<td></td>
<td>include 'see103hnj.pch'</td>
</tr>
<tr>
<td></td>
<td>ENDDATA</td>
</tr>
<tr>
<td></td>
<td>ENDDATA</td>
</tr>
<tr>
<td></td>
<td>ENDDATA</td>
</tr>
</tbody>
</table>
Example 5, using MATOP4 (or MATRIXOP4) storage option with two superelements:

**Step 1:**

**Superelement 1 creation**

ASSIGN OUTPUT4='s35212n.op4',unit=30,delete,format
SOL 109
CEND
Title = SE1 CREATE Example5
extseout(extbulk,extid=10,matop4=30)
SUBCASE 1
  disp = all
  stress = all
  strain = all
  force = all
BEGIN BULK
PARAM,USETPRT,0
aset1,123456,101,thru,104
CBEAM  1005    7  1001  1000  10000.
  ...
  ...
ENDDATA

**Superelement 2 creation**

ASSIGN OUTPUT4='s35212nar.op4',unit=31,delete
SOL 109
CEND
Title = SE2 CREATE Example5
extseout(asmbulk,extid=5060,matrixop4=31)
SUBCASE 2
  disp = all
  stress = all
  strain = all
  force = all
  mpcforce = all
BEGIN BULK
PARAM,USETPRT,0
aset1,123456,101,thru,104
CBAR  5005    6  5001  5005  10000.  10000.
  ...
  ...
ENDDATA
Step 2: System run with two superelements.

```verbatim
PROJECT='MODES'
ASSIGN INPUTT4='s35212n.op4',unit=30,format
ASSIGN INPUTT4='s35212nar.op4',unit=31
SOL 103
CEND
Title = SYSTEM RUN Example5
ECHO = sort
MAXLINES = 999999999
SUBCASE 1
  SUPER=10
  METHOD=11
  disp = all
  stress = all
  strain = all
  force = all
SUBCASE 2
  SUPER=5060
  METHOD=50
  disp = all
  stress = all
  strain = all
  force = all
  mpcf=force = all
SUBCASE 3
  SUPER=0
  METHOD=1
  SUPORT1=99
  disp = all
  stress = all
  strain = all
  force = all
  mpcf=force = all
BEGIN BULK
SEBULK  10  EXTOP4  MANUAL  30
SECONCT  10  0  NO
  101  101  102  102  103  103  104  104
include 's35212nar.asm'
SETREE,0,10,5060
EIGRL,1,-1.0, 80.
CBAR  2000  4  109  100  10000.
  ...
  ...
  ...
  ...
$ include 's35212n.pch'
$ include 's35212nar.pch'
$ EIGRL,11,-1.0,160.
EIGRL,50,-1.0,160.
ENDDATA
```
Chapter 3: Single-Level Superelement Analysis

- Introduction
- Simple Input for Single-Level Analysis
3.1 Introduction

We have defined single-level analysis as one in which all superelements connect to the residual structure only. Single-level superelement analysis occurs when the exterior points of all superelements in the model are interior to the residual structure. In this case, each superelement can be processed independently from all others.

This means that any boundary points where two or more superelements meet belong to the residual structure. Single-level analysis is the simplest form of superelement analysis and is recommended for the beginning superelement user. Partitioning the model into superelements for a single-level analysis requires the least effort from you, and processing control (if desired) is simple.

If PARTs are used, the model is automatically a single-level model unless either DT1, SETREE or SETREE entries exist in the Main Bulk Data Section. If the model is defined as a Main Bulk Data Section0—only model, you must verify that all exterior points belong to the residual structure; otherwise, a multilevel processing tree ("Multilevel Superelement Analysis") is created automatically.

In addition to single level being recommended for the beginning superelement user, single-level analysis is recommended for analyzing structures that are expected to change often and in many areas. For this case, where there is no knowledge of areas that will not be changing, a single-level model is almost always the most efficient. Consequently, when a change occurs, only the affected superelement and the residual structure need to be processed again. Therefore, restarts for model change can be very efficient. In fact, this is one of the biggest advantages of using superelements.

A good example is a spacecraft upper stage booster model used in support of a modal test. The model consisted of approximately 30,000 DOFs. To provide real time support to the test, modal runs incorporating changes in the model were necessary for evaluating how a change to the model might improve the match between test and analysis results. Initially, the entire model was being run for modes; this run took 29 CPU hours. (This was in 1985. It would run much faster on a current version of the software.) On a heavily-loaded computer, this run required about five days—obviously not real time turnaround. To improve the turnaround time, the model was divided into superelements based on an understanding of the structure and the areas expected to change. For this model, any areas that were expected to change were placed in the residual structure, and the rest of the model was placed in upstream superelements. Once this was done, an initial superelement run was performed. The initial run did not save any time (as expected). However, restart runs to evaluate the effects of a model change took only one CPU hour to run, including data recovery, a 29:1 performance improvement. Also, two runs could be processed in one day, compared to one run in a week without superelements.

3.2 Simple Input for Single-Level Analysis

This chapter contains the minimum information required to convert an existing model to superelements.

The simplest approach you can use for performing a superelement analysis is a single-level processing tree. In a single-level analysis, all points that are exterior to any superelement are interior to the residual structure. More simply, the points that are used to connect the superelements (exterior points) must belong to the residual structure. If any exterior point of any superelement belongs to another superelement, then the model is a multilevel model. This type of analysis is described later.

You are not required to use single- or multilevel analysis for any problem. You will discover, as you become a more advanced user, that some models lend themselves to a single-level analysis and others to a multilevel analysis. In statics, the two approaches provide identical answers. In
dynamic analysis, you may obtain different answers with a single-level solution than you would using a multilevel solution. This is discussed in the chapter on superelements in dynamic analysis.

**Single-Level Analysis Using Main Bulk Data Superelements**

Because you must define main bulk data superelements if you want to use a single-level processing tree, you must be sure that all exterior points belong to the residual structure. If any exterior points belong to another superelement, NX Nastran automatically creates a multilevel tree (if the model contains no PARTs).

For simplicity's sake, we will assume that an input file exists that can be used to perform a static solution on the model without superelements. That is, we will begin with an input file ready to run in a non-superelement static solution.

The input required to change from a conventional static analysis to a single-level superelement analysis is minimal. The steps are as follows:

1. Define the superelements. (To make a single-level model, make sure that the boundary points belong to the residual structure.)

2. Run using SOL 101.

These steps are the only requirements for changing a model from conventional analysis into single-level superelement analysis. (The format of the super case control command is covered in “Loads, Constraints, and Case Control in Static Analysis”.) If you only want to switch your model into a superelement model and run, you now have enough information.

However, a few limitations do exist. If you are using multiple boundary conditions, they must be contained in the residual structure. (The ability to solve with multiple residual structure boundary conditions was introduced in Version 68.) And if you are using inertia relief, there is an additional entry (PARAM,INREL,-1), which is described in “Inertia Relief Analysis Using Superelements”. For most static analysis input files, these requirements are all you need to convert from conventional to superelement analysis. The rest of this guide demonstrates advanced features and how to use superelements in dynamic analysis.

**Example of Single-Level Analysis**

For this example, we begin with an input file prepared for SOL 1 and convert it into a superelement analysis. The example problem shown in “Introduction and Fundamentals” is used. The SOL 1 input file is shown in Figure 3-1.
Figure 3-1. Conventional Static Analysis Input File.

This run has three loading conditions, which are described in “Introduction and Fundamentals”, and the entries for those loads are shown in Figure 3-2. The three loadings are:

- Load 101, the pressure load applied to the two square portions of the model.
- Load 201, point loads applied at the corners.
- Load 301, opposing point loads at the corners.
Now we will change this file into a superelement model. First we add the SESET entries (by including the file seset.dat from “How to Define a Superelement”). Once this is done, we change to SOL 101. The file shown in Figure 3-3 includes these changes (shown in bold). This file is ready to run and gives the same answers as those originally obtained in SOL 1.

```
PLOAD2,101,-1.,18,THRU,42
PLOAD2,101,-1.,43,THRU,67
$
$ LOAD CASE 2 - 2 POINT LOADS AT CORNERS
$
FORCE,201,93,,2.,0.,0.,1.
FORCE,201,104,,2.,0.,0.,1.
$
$ LOAD CASE 3 - OPPOSING POINT LOADS AT CORNERS
$
FORCE,301,93,,2.,0.,0.,1.
FORCE,301,104,,2.,0.,0.,-1.
$
```

Figure 3-2. File Load1.dat.
ID SE, SAMPLE PROBLEM SOL 101
$
$ SUPERELEMENT STATICS - SAMPLE PROBLEM - STATIC SOLUTION
$ USING SIMPLE CASE CONTROL
$

SOL 101 $ SUPERELEMENT STATICS - SINGLE LEVEL TREE
$ TIME 15
CBND
TITLE = S.E. SAMPLE PROBLEM 1
SUBTITLE = S.E. STATICS - RUN 1 - MULTIPLE LOADS
DISP = ALL
$
$
SUBCASE 101
LABEL = PRESSURE LOAD
LOAD = 101
$

SUBCASE 201
LABEL = 2# NORMAL LOADS
LOAD = 201
$

SUBCASE 301
LABEL = OPPOSING LOADS
LOAD = 301
$

BEGIN BULK
PARAM,POST,0
$

INCLUDE SESET.DAT
$

INCLUDE MODEL.DAT
INCLUDE LOAD1.DAT

Figure 3-3. File SE1S101.DAT.

Quick Review

Enough information has now been given to allow a single-level superelement static run to be performed using main bulk data superelements. Review the most important items listed below.

- Define the superelements to NX Nastran using field 9 on the GRID entry and/or SESET entries.
- NX Nastran automatically partitions the information in the Main Bulk Data Section for you.
- In case control, add SUPER = ALL above the first SUBCASE (optional—see the next chapter).
- Use SOL 101 for statics.
- The information given in this chapter is the minimum you need to begin using single-level superelement analysis. (For proficiency in static analysis, you should also read “Loads, Constraints, and Case Control in Static Analysis” through “Multiple Loadings in Static Analysis”.)
Single-Level Analysis When PARTs Are Present

When partitioned bulk data (PARTs) are used, NX Nastran automatically forces a single-level processing tree. As noted in the previous chapter, the program finds all grid points that are coincident between each PART and any other PARTs and/or main bulk data superelements, then (if the grid points are not already in the residual structure) creates internal copies of these points and places those copies in the residual structure. No special effort is required to run a single-level analysis if PARTs are present in the model. However, if you want to perform a multilevel analysis and PARTs are present, you must manually instruct the program to perform the multilevel solution (see “Multilevel Superelement Analysis”).
Chapter 4: Loads, Constraints, and Case Control in Static Analysis

- Introduction
- Loads in Static Analysis
- Boundary Conditions
- Case Control in Superelement Analysis
4.1 Introduction

A very simple first approach to using superelements in static analysis was presented in "Introduction and Fundamentals". Now we will go into more detail on how individual items work. For example, the SUPER = ALL command in “Single-Level Superelement Analysis” is the simplest form of the SUPER command, yet it has many additional features. Thus, “Single-Level Superelement Analysis” was the appetizer, and now we are prepared for the main course.

In this chapter, we discuss how loadings and constraints are partitioned when superelements are used and how they are applied in static analysis using case control commands.

4.2 Loads in Static Analysis

Loading entries in the Main Bulk Data Section are partitioned in the same manner as the grid points and elements. That is, loads applied on interior points of a superelement are placed into the bulk data for that superelement. Loads applied on exterior points are passed downstream. These loads are assigned to the superelement to which the grid point is interior. An exception is temperatures that are specified on an exterior point. These temperatures are copied into any superelements that use the grid point as exterior and the superelement that contains the grid point as interior.

Any loadings applied on an element are assigned to the superelement to which the element is interior. For PARTs, you must define the loading data within the partitioned bulk data (BEGIN SUPER=xx) for the PART.

Remember that each superelement has its own unique set of bulk data after partitioning. This means that loadings in one superelement cannot reference a grid point or element in another superelement. If they do, fatal errors result.

Note

Caution: Any loadings that use disjoint grid points or elements (that is, grid points or elements not interior or exterior to the current superelement) to define orientation result in fatal messages, because the location of these points and elements is not available to the current superelement. Also, loading entries using a THRU option should not reference elements in more than one superelement. Rather, a separate entry should be provided for each superelement. This problem does not arise when using most preprocessors, which generate one loading entry for each loaded element, but when entering data by hand, use caution.

Sample of Loading Data in the Main Bulk Data

The following example demonstrates the partitioning of loadings in the main bulk data (in superelement analysis):
Figure 4-1. Sample Loading in Superelement Analysis.

The above model, with the SESET entry shown, describes a single-level superelement problem with superelements 1 and 0. Superelement 1 contains interior grid points 4, 5, and 6. The residual structure (SEID=0) contains grid points 1, 2, and 3.

The bulk data entries for the applied loadings are partitioned as follows:

Force P1 is applied on grid point 6, which is interior to superelement 1; thus, force P1 is partitioned into the bulk data for superelement 1.

Force P2 is applied on grid point 3, which is interior to the residual structure; thus, force P2 is assigned to the residual structure bulk data.

pload W is applied on elements that belong interior to superelement 1; therefore, pload W is partitioned into the bulk data for superelement 1.

**Note**

The process just described is the partitioning of the bulk data for the applied loadings. The application of the loadings is actually controlled by the case control and is described later in this chapter.

Bulk data partitioning merely partitions the bulk data by superelement—it does not actually apply the loadings. The loadings are applied in the Case Control Section.

**Thermal Loadings in Superelement Analysis**

**Main Bulk Data Thermal Loads**

Temperatures described by grid points in the main bulk data (TEMP, TEMPD) are partitioned along with the grid points, with the following exception: Any temperatures defined at an exterior point are copied into both the superelement for which the grid point is exterior and the superelement for which the grid point is interior.

Temperatures defined by elements (TEMPPi and TEMPRB) are partitioned in the same manner as the elements.

**Partitioned Bulk Data Thermal Loads**

Once again, all temperature-related loading information for PARTs must be in the partitioned bulk data (BEGIN SUPER=xx) for the PART.
4.3 Boundary Conditions

Single-Point Constraints in Superelement Analysis

Single-Point Constraints in the Main Bulk Data

Any bulk data entries for single-point constraints are partitioned in a similar manner to the grid points. That is, an SPC entry referencing an interior point of a superelement is added into the bulk data for that superelement. An SPC entry referencing an exterior point is assigned to the bulk data of the superelement for which the grid point is interior, which is important for preparation of case control. That is, when applying constraints in case control, you must be careful to specify the correct set for each superelement. More details are provided later in this chapter.

SPC entries are partitioned in a manner similar to grid points. The entries are assigned to the bulk data for which the grid points they reference are interior.

The following sample helps demonstrate how SPC entries are partitioned.

![Diagram](Figure 4-2. Constraints in Superelement Analysis.)

The above model, with the SESET entry shown, describes a single-level superelement problem with superelements 1 and 0. Superelement 1 contains interior grid points 4, 5, and 6. The residual structure (SEID=0) contains grid points 1, 2, and 3.

Now we will look at how the constraint entries are partitioned.

Grid points 1 and 3 are interior to the residual structure; therefore, the constraint entries referencing them are assigned to the bulk data for the residual structure.

Grid point 6 is interior to superelement 1, and the SPC entry for grid point 6 is assigned to the bulk data for superelement 1.

Notice that although grid point 3 is exterior to superelement 1, the constraint is applied at the residual structure, which is important when specifying case control commands for this model.

Single-Point Constraints in Partitioned Bulk Data

SPC entries on a boundary of a PART are also applied to all coincident connection points from other PARTs.

Any constraint entries for PARTs must be contained within the partitioned bulk data for the part. If a constraint is to be applied to a boundary (exterior) point of a PART, then you must exercise caution.

When a constraint is defined on a boundary point, all six DOFs must be constrained on that grid point. Exceptions to this are as follows:
• If all coincident points use the same displacement coordinate system, then selected DOF may be constrained.

• Any constraints desired may be applied in the “downstream” superelement where the point is an interior point.

Grid Point Singularity Processor (PARAM, AUTOSPC, YES)

Because each superelement is processed independently, it is possible to turn AUTOSPC on and off, by superelement. Normally, you want to apply AUTOSPC to the entire model (all superelements), and that is the default in the superelement solutions (exception: the nonlinear solutions). However, the grid point singularity processor can be selectively applied to a model, using superelements, by either placing PARAM, AUTOSPC in the Case Control Section and using different values for each superelement (explained later in this chapter), or by specifying different values in the separate bulk data sections for PARTs.

When the grid point singularity processor (AUTOSPC) is used on superelements, it looks only at the interior points of each superelement. Any points that are exterior to a superelement are defined as belonging to the A-set for that superelement. AUTOSPC will not constrain an A-set DOFs. The exterior points of a superelement can have elements connected to them from other superelements; therefore, to constrain these points before we have a chance to attach all elements might overconstrain the structure. AUTOSPC looks at these grid points when processing the downstream superelement that contains them as interior points.

In the example used earlier in this chapter (Figure 4-1), when processing superelement 1, AUTOSPC could only apply constraints to singularities on grid points 4, 5, and 6. Any singularities on grid point 3 (which is exterior) were not constrained while processing superelement 1. Rather, they were constrained when processing the residual structure (which has grid point 3 interior).

MPCs and R-Type Elements in the Main Bulk Data Section

R-type elements (RBE2, RBE3, RBAR, RSSCON, etc.) in the main bulk data are treated as elements, that is, their connectivity is used to determine the superelement’s bulk data in which to put them and can be used to determine exterior points.

R-type elements are always included in the analysis. MPC entries are only included if the MPC set is called out in the Case Control Section for the superelement.

MPCs are treated in a similar manner, except that they are not used to define superelement exterior points. If we think back to “How to Define a Superelement”, we remember that element connectivity is used to determine the exterior points for a superelement. Because MPC sets are called in case control, and different MPC sets can be used in each SUBCASE, MPC sets are not used in the definition of superelements. MPC entries are partitioned according to the interior points of superelements. An MPC entry is assigned to the highest upstream superelement (based on the processing order) to which it connects.

Special care must be taken when an MPC or R-type element is connected to exterior points of a superelement. Because the exterior points are in the A-set for the superelement, these points cannot be listed as dependent (M-set) on an MPC or R-type element; otherwise a fatal message (UFM 2101A) results.

When MPCs or R-type elements reference an exterior point, the exterior point must always belong to the independent set for the entry.
The following example shows two ways in which a *rigid* connection between two superelements might be modeled. A typical situation calling for a rigid connection is when different groups have modeled different areas of the structure. When attaching the components together, MPCs or R-type elements are commonly used. In this case, superelement 1 consists of two CBAR elements connecting grid points 11, 12, and 13. The residual structure consists of two CBAR elements connecting grid points 1, 2, and 3. At the assembly level, grid points 3 and 11 are coincident and need to be attached. Figure 4-3 demonstrates two methods of making that attachment.

**Figure 4-3. Multipoint Constraints in Superelement Analysis.**

**Note**

Grid points 3 and 11 are coincident—they are shown offset for clarity.

**Method 1-RBAR**

RBAR,100,3,11,123456

This RBAR *connects* grid points 3 and 11, with grid point 3 as the independent grid point. Because RBAR 100 is treated as an element and connects to grid point 11 (which is interior to superelement 1), NX Nastran copies grid point 3 to the bulk data for superelement 1 (the most upstream superelement to which grid point 3 is connected) as an exterior point.

Notice that grid point 3 is the independent grid point on the RBAR. Once again, this is necessary, because grid point 3 is an exterior point and cannot be dependent on an R-type element or MPC in superelement 1.

**Method 2-MPCs**

The connection may also be made using multipoint constraints (MPCs). In this case, you need to be more careful because, as mentioned earlier, MPCs are not used in superelement bulk data partitioning. When partitioning the bulk data for superelement 1, NX Nastran automatically assigns the exterior points, based on element connectivity. A problem arises when the connection is made by an MPC, in that there is no connectivity to tell the program that grid point 3 should be exterior to superelement 1. If you do not make a special effort to do this, a fatal message results when the MPCs are processed for superelement 1.

This problem did not appear with the RBAR, because the RBAR is treated as an element for bulk data partitioning.

The following MPC entries are used to perform the connection:

MPC,1,11,1,1.,3,1,-1.
MPC,1,11,2,1.,3,2,-1.
These entries connect all DOFs of grid point 11 to grid point 3. Note that the coefficients are 1.0, because the points are coincident.

Grid point 3 is the independent grid point on the MPCs. Once again, an exterior point cannot be dependent on an MPC or R-type element.

Following are two possible methods of assigning grid point 3 as exterior to superelement 1 (to make the connection).

**Method 1, CSUPEXT**

CSUPEXT, 1, 3

This entry assigns grid point 3 as exterior to superelement 1. Although there is no connection made by elements, this point is now an exterior point, and the MPCs can be used to perform the connection.

**Method 2, PLOTEL**

NX Nastran treats PLOTELs the same as all other elements when partitioning bulk data; therefore, because this element has no stiffness associated with it, the PLOTEL is an ideal element for attaching exterior points to a superelement.

PLOTEL, 99, 3, 11

This element is assigned to superelement 1, because superelement 1 is the highest superelement in the processing tree that connects to this element. A plotel used with MPCs is also a device to symbolize MPCs on structure plots. When assigning exterior points to superelement 1, NX Nastran assigns grid point 3 as exterior, because this point is connected to PLOTEL number 99.

**MPCs and R-Type Elements in PARTs**

Because PARTs have separate Bulk Data Sections, any MPC entries and R-Type elements within a PART apply only to that PART. Note that, once again, exterior (boundary) points cannot be dependent on MPC equations or R-Type elements.

### 4.4 Case Control in Superelement Analysis

There are a number of superelement-unique case control commands. These commands include manual processing commands such as SEMG, SELG, SEKR, SELR, SEMR, SEALL, and SEDR. With the exception of SEDR, the details of these commands are not covered in detail in this section. They are covered in detail in a later chapter.

Information in this section refers to the Structured Solution Sequences (SSS), SOLs 100 through 200. The manual processing commands are not of particular interest in these solutions, because these solutions have automatic processing and restarts.

Case Control commands that are unique to superelement analysis are SEEXclude (SEEX), SEFINAL, and SUPER (the most important case control command for superelements). The SEFINAL and SEEX commands are processing commands that control processing order (SEFINAL) and connectivity (SEEX). The SUPER command is the command used to partition the case control into a unique subset for each superelement.
The SUPER Command-Case Control Partitioning

The SUPER command is used to partition the case control into a unique subset for each superelement.

The SUPER command is the most important case control command for superelements. It is also the most frequently misused, and when it is misused, either incorrect results or no output can occur. Therefore, you must pay strict attention to the use of the SUPER command.

As mentioned previously, if consistent input has been used, for most cases the SUPER=ALL command above the first SUBCASE is all that is needed. SUPER=ALL is the default. The following information is for the cases when different loadings, output requests, or processing requests are used for individual superelements.

If SUPER commands are used and a superelement is not mentioned in case control using the SUPER command, no loadings, SPCs, or MPCs are applied to that superelement, and no output is generated for it.

The purpose of the SUPER command is to partition the case control in a manner similar to the bulk data partitioning. That is, each superelement has its own unique set of case control created, based on the use of the SUPER command. When performing the processing for a superelement, the program only looks at the case control unique to that superelement. (Exception: Residual structure case control commands are used to determine the number of loading conditions being analyzed.) Thus, only the case control that has been specifically defined as applying to the superelement is available. If you have used a SUPER command and no case control commands that apply to a superelement are available, no loadings, SPCs, or MPCs are applied to that superelement and no output is generated for that superelement.

If no SUPER command occurs, the program assumes that the case control refers to all superelements. The form of the SUPER command is as follows:

\[ \text{SUPER} = I, J \]

where:

\( I \) = Superelement id or the id of a set of superelements. (If a set is used, the set id must be unique with respect to all superelement ids). The current default for the SUPER command is SUPER=ALL. Previously, the default was SUPER=0. Therefore, if old input files are used, the results may change.

\( J \) = Load sequence number (default = incremented starting with 1).

The SUPER command can occur in each SUBCASE and can appear before the first SUBCASE (in which case, it is a default).

The purpose of the SUPER command is to allow different loadings, constraints, and output requests for individual superelements.

If a SUPER command exists in your file, the default value for the SUPER command becomes SUPER = 0, which is backward compatible with prior versions of the software.
The easiest way to use the SUPER command is to either use the default, or to place SUPER = ALL before the first SUBCASE. In this case, all subcases apply to all superelements, and the case control looks similar to that for non-superelement solutions.

In superelement analysis, case control is treated differently than in conventional analysis. Internally, NX Nastran creates a separate case control set for each superelement (called CASES); thus, each superelement can have its own unique SUBCASE for each loading condition.

A SUBCASE must exist for the residual structure for each loading condition. An internal counter (J) is based on the sequence of residual structure SUBCASEs, which is in contrast with conventional analysis, where each SUBCASE defines a unique loading condition (or solution).

Processing-related commands (MPC, SPC, PARAM) must appear in the first SUBCASE referencing a superelement.

The intention is to give you flexibility, but the result is often confusion. One very confusing aspect is that any processing-related commands (MPC, SPC, and PARAMeters) must occur in the first SUBCASE for a superelement. The value of these commands must be the same in any additional SUBCASEs for that superelement. The exception is the residual structure, where MPC and SPC sets can be different in each static SUBCASE to account for multiple constraint conditions.

To demonstrate (and hopefully clarify), we start with a sample using conventional case control and explain the structure, followed by some examples of superelement case control.

**Conventional Case Control**

Expanded

```plaintext
SUBCASE 10  
  SET 1 = 101, THRU, 110  
  DISP = 1  
  LOAD = 100  
SUBCASE 20  
  SET 1 = 101, thru, 110  
  DISP = 1  
  LOAD = 200  
SUBCASE 30  
  SET 3 = 201, THRU, 210  
  DISP = 3  
  LOAD = 300
```

In the previous conventional case control, three loading conditions are defined. The first loading condition is defined by SUBCASE 10. In this SUBCASE, displacement output is requested for grid points 101 through 110, and loading set 100 is applied. The second loading condition is defined by SUBCASE 20. In this SUBCASE, displacement output is requested for grid points 101 through 110 again, but loading 200 is applied. Loading condition three is defined by SUBCASE 30. In this SUBCASE, loading set 300 is applied and output is requested for grid points 201 through 210.

The following condensed Case Control commands accomplish the same results.

**Condensed Case Control**

Anything specified before the first SUBCASE is a default for all SUBCASEs.

```plaintext
$  
SET 1 = 101, THRU, 110  
LOAD = 200  
SET 3 = 201, THRU, 210
```


DISP = 1
$
SUBCASE 10
  LOAD = 100
SUBCASE 20
SUBCASE 30
  LOAD = 300
  DISP = 3
$

In this condensed form, shared definitions and requests are placed before the first SUBCASE. (Anything that appears before the first SUBCASE is a default and is used for all SUBCASEs, unless specifically changed within a SUBCASE). If a default value is changed within a SUBCASE, the change is used only for that SUBCASE. Therefore, as before, SUBCASE 10 requests loading set 100 to be applied and displacement output to be calculated and printed for grid points 101 through 110. Subcase 20 requests displacement output for grid points 101 through 110 while applying load set 200. Subcase 30 requests displacement output for grid points 201 through 210 while applying load set 300. These results are identical to those from the expanded case control.

**Superelement Case Control**

In superelement case control, the same rules apply as in conventional case control, with the exception that each superelement can have a SUBCASE defined for each loading condition. We first look at the superelement case control for a model with only one loading condition, then we move up to multiple loading conditions.

**One Loading Condition—Expanded Case Control**

For this example, we use the model in Figure 4-1, which contains one superelement and the residual structure and has two loading conditions defined. For this example, we assume that loadings P1 and W belong to loading set 1, which is the applied loading. For now, we do not apply load P2.

```plaintext
SUBCASE 1
  SUPER = 1
  LOAD = 1
  DISP = ALL
SUBCASE 2
  SET 99 = 0
  SUPER = 99
  LOAD = 1
  DISP = ALL
```

Loading condition one consists of two SUBCASEs: SUBCASE 1 and 2. SUBCASE 1 tells NX Nastran to apply any loadings from set 1 that occur in its bulk data to superelement 1 and to print displacements for all grid points in the bulk data for superelement 1. SUBCASE 2 tells NX Nastran to do the same for the residual structure.

Notice that in this case the load sequence number is not used, which causes NX Nastran to assume that the first SUBCASE for a superelement is to be used on loading condition 1, the second SUBCASE for that superelement applies to loading condition 2, etc.

Using this information, SUBCASE 1 applies to superelement 1 for loading condition 1, and SUBCASE 2 applies to the residual structure for loading condition 1. A way to visualize this is as follows: The program creates a separate set of case control for each superelement, based on the input. When processing each superelement, the program only looks at the associated set of case control. For the above example, you could think of the case control as being divided into two sets, one for superelement 1, and the other for the residual structure. (The example given below is shown to
help you understand how this process works, but remember that this process is done internally with binary tables.)

Case Control for superelement 1
SUBCASE 1
SUPER = 1
LOAD = 1
DISP = ALL

Case Control for the residual structure
SUBCASE 2
SUPER = 0
LOAD = 1
DISP = ALL

When the program is processing superelement 1, it looks at the case control for superelement 1 and sees that there is only one solution to be generated. For that solution, load 1 is applied, and displacement output is requested for all grid points in the superelement.

When processing the residual structure, the program sees that there is only one solution to be generated, generates matrices for any loading entries corresponding to set 1 in the residual structure, and combines that information with the reduced loading matrices from the upstream superelement.

For each solution (Loading Sequence) defined, there is one column in the loading matrix. The number of solutions (columns in the loading matrix) is determined by the number of residual structure SUBCASEs. Each superelement generates a loading matrix with that number of columns containing information based on the case control for that superelement. If a superelement does not have a SUBCASE associated with a residual structure SUBCASE (Loading Sequence), a null column is generated in the loading matrix for that superelement for that solution.

One Loading Condition-Condensed Case Control

Now we show case control for the same solution using the condensed form, which is closer to what most of us are accustomed to seeing.

SUPER = ALL $ optional in MSC.Nastran V69+, necessary in previous versions
DISP = ALL
LOAD = 1

Because only one loading condition is applied, the default of having all case control apply to all superelements is used. A SUBCASE command could be added to this case control, but it is not required if there is only one loading condition.

Now that we have shown the simplest use of the SUPER command, we look into possible options.

First, let us look at the possibility of applying different loadings and requesting different output for each superelement with only one loading condition. We assume that we have had different groups model the structure. We now have to assemble the pieces into one model and perform the solution. We will look at the two-headed flyswatter shown in Figure 1-12, using the superelements shown in Figure 2-1.

We write case control to apply the pressure loading, assuming that different groups modeled each superelement, where the group modeling superelement 1 used loading set 101, and the group modeling superelement 2 used loading set 102 to represent the pressure loading.

Also, the group modeling superelement 3 has requested that we provide grid point force balance output and displacements, while all others have requested displacement and stress output.

The Case Control for this appears as follows:

DISP = ALL $ default request before first SUBCASE
STRESS = ALL $ default request before first SUBCASE
SET 99 = 0,4,5,6,7 $ set of superelements
SUBCASE 10
  LABEL = SUPERELEMENT 1, LOADING 1
  SUPER = 1
  LOAD = 101
SUBCASE 20
  LABEL = SUPERELEMENT 2, LOADING 1
  SUPER = 2
  LOAD = 102
SUBCASE 30
  LABEL = SUPERELEMENT 3, LOADING 1
  SUPER = 3
  GPFORCE = ALL
  STRESS = NONE
SUBCASE 40
  LABEL = SUPERELEMENTS 0,4,5,6,7, LOADING 1
  SUPER = 99

In this set of case control, we have defined a set containing superelements 4, 5, 6, 7, and the residual structure. We have also stated that, by default, we would like to obtain displacement and stress output for all SUBCASEs. SUBCASEs 10 and 20 are used to apply the loading on superelements 1 and 2 respectively and to obtain the output for these superelements. SUBCASE 30 is used for superelement 3. Although this SUBCASE does not have any loadings applied directly, it must be mentioned in case control to obtain output. In this SUBCASE, displacement and grid point force balance output have been requested, and the default stress output request has been overridden.

The final SUBCASE is for the remaining superelements and the residual structure. Once again, it is mandatory to have a SUBCASE for the residual structure for all loading conditions. This final SUBCASE satisfies that requirement. Also, to obtain output for a superelement, that superelement must be mentioned in the case control. The following case control, which has set 99 changed to reference the residual structure only, solves the problem for the same loadings but has no output for superelements 4, 5, 6, or 7.

DISP = ALL $ default request before first SUBCASE
STRESS = ALL $ default request before first SUBCASE
SET 99 = 0 $ set of superelements
SUBCASE 10
  LABEL = SUPERELEMENT 1, LOADING 1
  SUPER = 1
  LOAD = 101
SUBCASE 20
  LABEL = SUPERELEMENT 2, LOADING 1
  SUPER = 2
  LOAD = 102
SUBCASE 30
  LABEL = SUPERELEMENT 3, LOADING 1
  SUPER = 3
  GPFORCE = ALL
  STRESS = NONE
SUBCASE 40
  LABEL = SUPERELEMENTS 0,4,5,6,7, LOADING 1
  SUPER = 99

By removing superelements 4, 5, 6, and 7 from set 99, we have removed any reference from the case control for superelements 4, 5, 6, and 7. These superelements are processed and their stiffness is included in the solution, but no loadings are applied to them (acceptable for this case) and no output is available for them.
Case Control for Multiple Loading Conditions

Once again, if the input is coordinated, the case control for multiple loading conditions looks almost identical to conventional case control. For example, using the example model in Figure 1-12, divided into superelements as shown in Figure 2-1, we will demonstrate how the case control looks using superelements for multiple loading conditions.

Case 1: Simplified Case Control

The following case control is the simplified (or condensed) case control for multiple loading conditions. This case control is based on using the file load1.dat, as shown in Figure 4-5, to define the input loadings. For this case, the model is contained within the Main Bulk Data Section only. If PARTs were used, then all loading entries for each PART would have to be in the correct BEGIN SUPER section of the input file.

```
$ Set DEFAULT - ALL CASE CONTROL IS FOR ALL SUPERELEMENTS
SUPER = ALL $ not needed, but it is better to include this
SUBCASE 101
LABEL = PRESSURE LOAD
LOAD = 101
$
SUBCASE 201
LABEL = 2# NORMAL LOADS
LOAD = 201
$
SUBCASE 301
LABEL = OPPOSING LOADS
LOAD = 301
$

$ Set DEFAULT - ALL CASE CONTROL IS FOR ALL SUPERELEMENTS
SUPER = ALL $ not needed, but it is better to include this
SUBCASE 101
LABEL = PRESSURE LOAD
LOAD = 101
$
SUBCASE 201
LABEL = 2# NORMAL LOADS
LOAD = 201
$
SUBCASE 301
LABEL = OPPOSING LOADS
LOAD = 301
$
```

Figure 4-4. Simplified Case Control for Multiple Loadings.

This case control is similar to the conventional case control for multiple loadings. The only change is that a SUPER = ALL command has been added to indicate that the case control applies to all superelements. Note that this is now the default and therefore is not necessary.

Notice that in Figure 4-5, each loading condition has the same load set id for all loaded superelements. Notice the PLOAD2 entries. Although elements 18 through 67 have the same load, the loading has been entered using two PLOAD2 entries because, once again, the bulk data is partitioned by superelement. If a single PLOAD2 entry was used referencing the entire range, a copy of that
entry would be made for both superelements 1 and 2, possibly resulting in a fatal message being issued when processing the loadings for the superelements, because neither superelement has the complete range of elements. Superelement 1 has elements 18 through 42; therefore, the program would issue fatal messages stating that elements 43 through 67 did not exist.

Thus, splitting the PLOAD2 into two entries as shown prevents problems. Most preprocessors generate a PLOAD2 for each element, which prevents this problem from occurring.

```
FORCE,301,93,,2.,0.,0.,1.
FORCE,301,104,,2.,0.,0.,-1.
$
```

```
FORCE,301,93,,2.,0.,0.,1.
FORCE,301,104,,2.,0.,0.,-1.
$
```

Figure 4-5. File load1.dat.

Case 2: Expanded Case Control

We now use the loading information from the file load2.dat, shown in Figure 4-6. In this file, each loading condition is represented by two load sets. For example, loading condition 1, the pressure loading, is represented by load sets 101 and 102.

```
$
$ LOADINGS - FOR RUN SHOWING PARTITIONED CASE CONTROL
$
$ LOAD CASE 1 - PRESSURE LOAD
$
PLOAD2,101,-1.,18,THRU,42
PLOAD2,102,-1.,43,THRU,67
$
$ LOAD CASE 2 - 2 POINT LOADS AT CORNERS
$
FORCE,201,93,,2.,0.,0.,1.
FORCE,202,104,,2.,0.,0.,1.
$
$ LOAD CASE 3 - OPPOSING POINT LOADS AT CORNERS
$
FORCE,301,93,,2.,0.,0.,1.
FORCE,302,104,,2.,0.,0.,-1.
$
```

Figure 4-6. File load2.dat.

The problem of combining these load sets together could be overcome by using the LOAD Bulk Data entry to instruct the program to combine the loadings. However, as we look at the Case Control, we use Case Control to determine how the loadings will be applied. The case control commands that combine the loadings are shown below.
Figure 4-7. Expanded Case Control for Multiple Loadings.
In this case control, default values have been set before the first SUBCASE. By default, any SUBCASE without a SUPER command applies to SET 999 (SE 0, 3, 4, 5, 6, 7), and the default output request for any SUBCASE is displacements for all grid points.

Load condition 1 is defined by three SUBCASEs: SUBCASEs 101, 102, and 1001. In SUBCASE 101, the command SUPER = 1,1 tells the program that this SUBCASE is for superelement number 1, as well as for defining the actions to be used in load sequence 1 (in this case, to apply load set 101).

SUBCASE 102 has the command SUPER = 2,1. This command tells the program this SUBCASE contains the commands for superelement 2 in loading sequence 1 (in this case, apply load set 102).

SUBCASE 1001 has no SUPER command, thus NX Nastran uses the default assigned before the first SUBCASE, which applies to superelements 0, 3, 4, 5, 6, and 7. Because there is no load sequence number provided, the program assigns a load sequence number based on the order in which the SUBCASE appears. Because this is the first SUBCASE for these superelements, this SUBCASE is used for loading sequence 1, just as if we had entered the command SUPER = 999,1. No loadings are applied in this SUBCASE, so the only loadings for this loading condition are those applied to superelements 1 and 2.

Loading condition 2 is defined in a similar manner using SUBCASEs 201, 202, and 1002. Loading condition 3 is defined by SUBCASES 301, 302, and 1003.

**Note**

Be careful—if no SUBCASE exists for a superelement for a loading, no loads are applied to that superelement for that loading, and no output is available for that superelement for that loading.

A common omission in the Case Control Section is to forget to include a command for a superelement that has no loads or constraints applied to it. This omission does not cause incorrect answers but prevents you from obtaining any output for that superelement for that loading condition.

**Note**

This condition can be corrected by adding a SUBCASE for the superelement on a restart run.

In this example, SUBCASEs 1001, 1002, and 1003 have been used to define the output requests for superelements 0, 3, 4, 5, 6, and 7. Note that if the SUPER=999 command had been omitted, these SUBCASEs would apply only to the residual structure, and therefore would not request any output for superelements 3, 4, 5, 6, or 7.

**Second Example of Expanded Case Control**

We will assume that the following model has been created by two separate engineering groups. One group (group A) has created the model for superelement 1. This group used load set 1 to define the first loading condition and loading set 2 to define the second loading condition. Another group (group B) has created the model that will be used as the residual structure. Unfortunately, due to a lack of communication, group B used load set 2 to define loading condition 1 and load set 1 to define loading condition 2.

In a conventional analysis, this problem could not be solved without making changes to the bulk data, with the ensuing potential for error. Using the expanded case control and superelements allows the original bulk data to be used without modification.
Figure 4-8. Sample Model Requiring Expanded Case Control.

```
CBAR  1  1  1  2  1.
CBAR  2  1  2  3  1.
G     1  0.0  0.0  0.0
G     2  2.0  0.0  0.0
G     3  4.0  0.0  0.0
PLOAD1  1  1  FY  FR  .5  -1.
PLOAD1  2  2  FY  FR  .3  -1.
SPC  1  1  123456  0.0
PBAR  1  1  1  10.  10.  10.
MAT1  1  1.7  .3  .1
BEGIN SUPER=1
G     3  4.0  0.0  0.0
G     4  6.0  0.0  0.0
G     5  8.0  0.0  0.0
G     6 10.0  0.0  0.0
CBAR  3  1  3  4  1.
CBAR  4  1  4  5  1.
CBAR  5  1  5  6  1.
FORCE  1  6  0  1.  -1.
PLOAD1  1  4  FY  FR  0.0  -1.  1.  -1.
PLOAD1  1  3  FY  FR  0.0  -1.  1.  -1.
PBAR  1  1  1  10.  10.  10.
MAT1  1  1.7  .3  .1
```

Figure 4-9. FILE EXP.DAT - Bulk Data for Expanded Case Control Example.
For this model, we define grid points 4, 5, and 6 as being interior to superelement 1, and the rest of the grid points belong to the residual structure. Now the following input file results in the correct answers.

```
ID       SAMPLE, EXPANDED CASE CONTROL
SOL      101  $
TIME     5
CEND
TITLE = SAMPLE INPUT REQUIRING EXPANDED CASE CONTROL
DISP = ALL
SET 99 = 0
SUBCASE 1
    SUPER = 1,1
    LOAD = 1
SUBCASE 2
    SUPER = 99,1
    SPC = 1
    LOAD = 2
SUBCASE 11
    SUPER = 1,2  $ no applied loads -
    $ this SUBCASE is simply to obtain output
SUBCASE 12
    SUPER = 99,2
    LOAD = 1
    SPC = 1
BEGIN BULK
INCLUDE 'exp.dat'
ENDDATA
```

**Figure 4-10. Input File for Expanded Case Control Sample.**

The expanded case control shown above applies the loadings as desired. SUBCASEs 1 and 2 are used to describe the first loading condition. In these SUBCASEs, load set 1 is used for superelement 1, and load set 2 is used for the residual structure.

Loading condition 2 is described using SUBCASEs 11 and 12. In these SUBCASEs load set 1 is applied on the residual structure, and no loadings are applied on superelement 1.

SUBCASE 11 has no loading request, because there is no load applied on superelement 1 in the second loading condition. The purpose of this SUBCASE is to obtain output for superelement 1 from the second loading condition. If this SUBCASE did not exist, then there would be no output for superelement 1 for the second loading, although its stiffness would be included in the solution.

**Parameters in the Case Control Section**

With NX Nastran, you can specify parameters in case control. Many—but not all—parameters can be specified in case control. In most cases, these parameters can have a unique value for each superelement. The parameter K6ROT is an example of this ability. This parameter is used to add a fictitious stiffness for in-plane rotational stiffness for CQUAD4 and CTRIA3 elements. You might utilize this feature in only a limited part of the model or might wish to use different values for different areas. Because you can specify values in case control, you can specify a different value for each superelement.
PARAM entries can be specified above the subcase level or within individual subcases. In all solutions except nonlinear, the value used in the first subcase appearing for a superelement is used for that superelement for all subcases.

If a parameter appears above the first subcase, the value given is a default value and is used for all subcases (if not changed inside individual subcases).

The value used for any parameter when processing a superelement is determined with a strict set of rules. These rules state that the value used for a parameter is:

- The value from the first subcase for that superelement.
- The default value specified before the first subcase (if not in the subcase).
- The value specified in the bulk data (if not either of the above).
- The default value of the parameter (if none of the above). Note that the default value can vary between solutions.

Due to this hierarchy, if you want to use different values for parameters for different superelements, we recommend that you specify a default value before the first subcase and specify any exceptions inside the appropriate subcases. Another option, which is even more conservative, is to specify the correct parameter value inside each subcase.

```
TITLE = PARAMETERS IN CASE CONTROL
PARAM,GRDPNT,100
SUBCASE 1
SUPER = 1
PARAM,GRDPNT,0
...
SUBCASE 2
SUPER = 2
...
SUBCASE 3
SUPER = 3
PARAM,GRDPNT,200
...
SUBCASE 4
SUPER = 4
...
SUBCASE 100 $ RESIDUAL STRUCTURE
SUPER = 0
PARAM,GRDPNT,-1
...
BEGIN BULK
```

**Figure 4-11. Sample of Parameters in Case Control.**

The above sample illustrates the use of parameters in the Case Control Section. In this case, different values are used for the parameter GRDPNT in the grid point weight generator calculation for each superelement.
In this case control, a default value of 100 is set for parameter GRDPNT by placing it before the first subcase. If we go through the case control by subcase, we can see which value is used for GRDPNT for each superelement.

The first subcase applies to superelement 1. Parameter GRDPNT is given a value of 0 in this subcase; therefore, the grid point weight generator uses the basic origin (if GRDPNT=0, then the basic origin is used) as the reference grid point for superelement 1.

The second subcase applies to superelement 2. In this subcase, there is no value provided for GRDPNT; therefore, the default value given before the first subcase is used. The grid point weight generator for superelement 2 uses grid point 100 as the reference point. Note that if grid point 100 is not included in the partitioned bulk data for superelement 2, then information message 3041 is issued by the program and the basic origin is used.

The third subcase applies to superelement 3. Once again, a value is provided for GRDPN in this subcase. The grid point weight generator output for superelement 3 uses grid point 200 as the reference grid point. (Once again, if grid 200 is not in the partitioned data for superelement 3, the basic origin is used.)

The fourth subcase applies to superelement 4. No value is provided for GRDPNT in this subcase; therefore, the default value of 100 is used.

The final subcase applies to the residual structure. In this subcase, a value of -1 is provided for GRDPNT, meaning that the grid point weight generator is not calculated for the residual structure.

**Note**

When processing superelements, the output from Grid Point Weight Generator contains the current superelement only. Assembly weight is not printed.
Chapter 5: Inertia Relief Analysis Using Superelements

- Introduction
- The Concept of Inertia Relief
- Interface for Inertia Relief Using Superelements
- Example of Inertia Relief Using Superelements
5.1 Introduction

Inertia relief is a subset of static analysis. Inertia relief is normally used to allow static solution of models that are not stable. That is, if a structure has possible mechanisms or rigid body motion, a static analysis cannot be performed using conventional methods. Inertia relief provides an approach that can solve problems containing mechanisms or rigid body modes.

Refer to the following reference documentation for further description and theoretical discussion:

- “Inertia Relief in Linear Static Analysis” in the *NX Nastran User’s Guide*.
- “Solution Sequences” in the *NX Nastran User’s Guide*.

5.2 The Concept of Inertia Relief

When using inertia relief, you want to obtain a static solution for a structure that is not stable, but is in a state of equilibrium under loading. Examples of structures in this state are aircraft in flight, cars in motion, and spacecraft in orbit. Each of these structures has one or more unconstrained, rigid body motions possible. Therefore, it is not possible to solve these problems using conventional static analysis. In inertia relief, the structure is in a state of static equilibrium under applied loading and inertial loads.

When inertia relief is invoked, the program calculates the summation of all loads and the mass properties about the reference degrees-of-freedom (DOFs), and then the inertial loads required to put the structure in a state of static equilibrium are calculated. The program applies the static loads and these accelerations simultaneously. The model is now in a state of equilibrium, and a set of constraints must be added that can prevent any rigid body motion or mechanisms. The program accomplishes this task by constraining the reference DOFs and performing the solution. If the structure is capable of rigid body motion or mechanisms and the reference DOFs can prevent these, then the reaction forces at the reference DOFs will be zero.

The implementation in SOL 101 (superelements) allows you to specify a maximum of six reference DOFs. The intent is to allow the solution of structures that are not constrained (or to allow rigid body motion) and not mechanisms. At the superelement level, SOL 101 calculates additional loadings (beyond those specified in the Case Control Section). Each of these loads corresponds to a unit acceleration of the superelement mass about the reference point (PARAM,GRDPNT). The procedure used to solve the inertia relief problem is described in the “Solution Sequences” in the *NX Nastran User’s Guide*.

5.3 Interface for Inertia Relief Using Superelements

In SOL 101, inertia relief is invoked using a SUPORT entry and PARAM,INREL,−1. If a SUPORT entry is used and PARAM,INREL is not set to −1, the program performs a static solution with the reference (SUPORT) DOFs constrained and does not calculate the acceleration loads required for equilibrium.

PARAM,GRDPNT, if used in a run using inertial relief, must be specified for all superelements, and the location of the reference grid point must be identical for all superelements.

An additional Bulk Data entry, DMIG,UACCEL, is available so user-specified accelerations can be used in place of the ones calculated for equilibrium. If this entry is used, you must specify the point...
about which the accelerations will be applied (this point must appear on a SUPORT entry), and you must specify the accelerations to be applied for each solution (SUBCASE).

The SUPORT entry, PARAM,INREL,−1, and the DMIG UACCEL entries must be in the Main Bulk Data Section, and they must point to grid points that are in the residual structure.

If PARTs are used for inertia relief, PARAM,INREL,−1 must appear in the bulk data for each PART and in the main bulk data.

PARAM,GRDPNT can be used to provide the location to sum the mass properties when calculating the accelerations, but if PARAM,GRDPNT is used, it should be specified for each superelement, and the points referenced on it should be in the same location. If GRDPNT references a point that is not in a superelement, the program terminates with User Fatal Message 9006.

5.4 Example of Inertia Relief Using Superelements

For this example, the two-headed flyswatter model will be used, with the three static loadings described in “Introduction and Fundamentals”. Because the goal of this problem is to demonstrate inertia relief, the model will be analyzed as free-free by removing the constraints on grid points 1 and 2 and all constraints on the z-rotation. Use PARAM,K6ROT to add in-plane rotational stiffness to the elements so that a single point can be used as the SUPORT. Grid point 13 (which lies in the residual structure close to the geometric center of gravity) is chosen for the reference point. The model will be run three different ways:

1. Without superelements (reference run).

2. Using PARTs.

3. Using superelements in the main bulk data only.

All three models result in the same solution. Selected output from file inrel2.f06 is included after the input file listings. Notice that PARAM,INREL,−1 must be specified for each superelement; therefore, in the model using PARTs, PARAM,INREL,−1 must be specified in each PART’s input data.

```$ file - inrel1.dat
$ inertial relief - without using superelements
$ inertial relief - note PARAM,INREL,1
$ GRDSET removed and K6ROT is set in case control
$ constraints removed on grids 1 & 2
$-----------------------------------------------
SOL 101
TIME 15
CEND
TITLE = File inrel1.dat - inertia relief using main bulk data
SUBTITLE = reference point is point 13
DISP = ALL
stress = all
PARAM,K6ROT,1000.
SUBCASE 101
LABEL = PRESSURE LOAD
LOAD = 101
$
SUBCASE 201
LABEL = 2# NORMAL LOADS```
LOAD = 201
$ $ SUBCASE 301
LABEL = OPPOSING LOADS
LOAD = 301
$ $
BEGIN BULK
$ $ unique entries for inertia relief
$ $ SUPORT,13,123456
PARAM,INREL,-1
$ param,post,-1
$ include ’model.dat’
include ’load.dat’
$
enddata

Listing 5-1. inrel1.dat

$ file - inrel2.dat
$ $ all 7 s.e. brought in using begin super
$ duplicate boundary grids id
$ each s.e. contains its own property description with the same id
$ condensed subcase setup
$ $ inertia relief - note PARAM,INREL,-1 is in all partitioned Bulk Data sections
$ include files have GRDSET removed and K6ROT is set in Case Control
$ constraints removed on grids 1 & 2
$ $
$ $ -----------------------------------------
$
SOL 101
TIME 15
CEND
TITLE = File inrel2.dat - inertia relief using PARTs
SUBTITLE = reference point is point 13 in residual
DISP = ALL
stress = all
SET 999 = 0,1,2,3,4,5,6,7
SUPER = 999 $ ALL CASE CONTROL IS FOR ALL SUPERELEMENTS
PARAM,K6ROT,1000.
SUBCASE 101
LABEL = PRESSURE LOAD
LOAD = 101
$
$ SUBCASE 201
LABEL = 2# NORMAL LOADS
LOAD = 201
$
$ SUBCASE 301
LABEL = OPPOSING LOADS
LOAD = 301
$
$
BEGIN BULK
$ $ unique entries for inertia relief
$ $ SUPORT,13,123456
PARAM,INREL,-1
$
Inertia Relief Analysis Using Superelements

param, post, -1

CQUAD4 5 1 13 14 24 23

GRID 13 -.4 3.6 0.
GRID 14 .4 3.6 0.
GRID 23 -.4 4.4 0.
GRID 24 .4 4.4 0.

include 'prop1.blk'
begin super=1
PARAM, INREL, -1
include 'loadse1.blk'
include 'prop1.blk'
include 'se1i.blk'
begin super=2
PARAM, INREL, -1
include 'loadse2.blk'
include 'prop1.blk'
include 'se2i.blk'
begin super=3
PARAM, INREL, -1
include 'prop1.blk'
include 'se3i.blk'
begin super=4
PARAM, INREL, -1
include 'prop1.blk'
include 'se4i.blk'
begin super=5
PARAM, INREL, -1
include 'prop1.blk'
include 'se5i.blk'
begin super=6
PARAM, INREL, -1
include 'prop1.blk'
include 'se6i.blk'
begin super=7
PARAM, INREL, -1
include 'prop1.blk'
include 'se7i.blk'
enddata

Listing 5-2. inrel2.dat

file - inrel3.dat

all 7 s.e. in main bulk data section

inertia relief - note PARAM, INREL, -1
$ GRDSET removed and K6ROT is set in case control
$ constraints removed on grids 1 & 2
$ 
$ SOL 101
TIME 15
CEND
TITLE = File inrel1.dat - inertia relief using main bulk data
SUBTITLE = reference point is point 13 in residual
DISP = ALL
stress = all
SET 999 = 0,1,2,3,4,5,6,7
SUPER = 999 $ ALL CASE CONTROL IS FOR ALL SUPERELEMENTS
PARAM,K6ROT,1000.
SUBCASE 101
LABEL = PRESSURE LOAD
LOAD = 101
$
SUBCASE 201
LABEL = 2# NORMAL LOADS
LOAD = 201
$
SUBCASE 301
LABEL = OPPOSING LOADS
LOAD = 301
$
$
BEGIN BULK
$
$ unique entries for inertia relief
$
$ SUPORT,13,123456
PARAM,INREL,-1
$
$ param.post,-1
$
include ’model.dat’
include ’seset.dat’
include ’load.dat’
$
enda
data

Listing 5-3. inrel3.dat

$ model.dat
$
$ **************************************************************
$ BASIC MODEL DEFINITION - SAME FOR ALL RUNS
$ **************************************************************
$ $ FILE NAME IS MODEL.DAT
$ param,k6rot,100.
GRID,1,,-.4,0.,0.,123456
GRID,3,,-.4,0.9,0.
=,*2,=,*9,==
=1
GRID,2,,.4,0.,0.,123456
GRID,4,,.4,0.9,0.
=,*2,=,*9,==
=1
GRID,9,,3.6,3.6,0.
=,*1,=,*8,==
=8
GRID,19,-3.6,4.4,0.
=,*1,=*8,==
=8
GRID,29,-3.6,5.2,0.
GRID,30,-2.8,5.2,0.
GRID,31,2.8,5.2,0.
GRID,32,3.6,5.2,0.
GRID,33,-5.2,6.,0.
=,*1,=*8,==
=4
GRID,39,1.2,6.,0.
=,*1,=*8,==
=4
GRID,45,-5.2,6.8,0.
=,*1,=*8,==
=4
GRID,51,1.2,6.8,0.
=,*1,=*8,==
=4
GRID,57,-5.2,7.6,0.
=,*1,=*8,==
=4
GRID,63,1.2,7.6,0.
=,*1,=*8,==
=4
GRID,69,-5.2,8.4,0.
=,*1,=*8,==
=4
GRID,75,1.2,8.4,0.
=,*1,=*8,==
=4
GRID,81,-5.2,9.2,0.
=,*1,=*8,==
=4
GRID,87,1.2,9.2,0.
=,*1,=*8,==
=4
GRID,93,-5.2,10.,0.
=,*1,=*8,==
=4
GRID,99,1.2,10.,0.
=,*1,=*8,==
=4
$
$ ELEMENTS
$
$ CQUAD4,1,1,1,2,4,3
=,*1,=*2,*2,*2,*2
=1
CQUAD4,4,1,7,8,14,13
CQUAD4,6,1,9,10,20,19
=,*1,=*1,*1,*1,*1
=2
CQUAD4,5,1,13,14,24,23
CQUAD4,10,1,14,15,25,24
=,*1,=*1,*1,*1
=2
CQUAD4,14,1,19,20,30,29
CQUAD4,15,1,29,30,36,35
CQUAD4,16,1,27,28,32,31
CQUAD4,17,1,31,32,42,41
CQUAD4,18,1,33,34,46,45
=,*1,=*1,*1,*1
=3
CQUAD4,23,1,45,46,58,57
=,*1,=*1,*1,*1,*1
Listing 5-4. model.dat

FILE SESET.DAT

DEFINE S.E. MEMBERSHIP OF GRID POINTS FOR SINGLE-LEVEL SUPERELEMENT

SAMPLE PROBLEM

SESET,1,33,34,37,38
SESET,1,45,THRU,50
SESET,1,57,THRU,62
SESET,1,69,THRU,74
SESET,1,81,THRU,86
SESET,1,93,THRU,98
SESET,2,39,40,43,44
SESET,2,51,THRU,56
SESET,2,63,THRU,68
SESET,2,75,THRU,80
SESET,2,87,THRU,92
SESET,2,99,THRU,104
SESET,3,29,30
SESET,4,31,32
SESET,5,21,22
SESET,5,9,THRU,12
SESET,6,25,26
SESET,6,15,THRU,18
SESET,7,1,THRU,8
$
$
Listing 5-5. seset.dat
$
$
Listing 5-6. load.dat

Selected Output from inrel2.f06

The following contains selected output from the results of the inertia relief solution obtained using PARTs. Once again, all three runs provided the same results.

Individual Superelement Output

For each superelement, the output includes the OLOAD RESULTANT, which is described in the “Applied Loads Check” section in the *NX Nastran User’s Guide*. In addition to the resultant for each subcase, the result output includes six additional loading vectors. These six loading vectors represent the loadings resulting from applying a unit acceleration about the reference grid point (PARAM,GRDPNT), or the origin of the basic coordinate system, in each of the six DOFs. These loads are not applied directly. Rather, they are used to calculate the inertial loadings to put the applied loads into equilibrium. This calculation is performed as a part of the residual structure operations. (The actual matrix operations are described in “Solution Sequences” in the *NX Nastran User’s Guide*).

The resultant “TOTALS” from the output of each subcase is included below. The actual output also includes the FX, FY, FZ, MX, MY, and MZ rows.

<table>
<thead>
<tr>
<th>SUBCASE</th>
<th>T1</th>
<th>T2</th>
<th>T3</th>
<th>R1</th>
<th>R2</th>
<th>R3</th>
</tr>
</thead>
<tbody>
<tr>
<td>101</td>
<td>0.0000000E+00</td>
<td>0.0000000E+00</td>
<td>-1.6000000E+01</td>
<td>-1.2800000E+02</td>
<td>-5.1200000E+00</td>
<td>0.0000000E+00</td>
</tr>
<tr>
<td>201</td>
<td>0.0000000E+00</td>
<td>0.0000000E+00</td>
<td>2.0000000E+02</td>
<td>2.0000000E+01</td>
<td>1.0400000E+00</td>
<td>0.0000000E+00</td>
</tr>
<tr>
<td>301</td>
<td>0.0000000E+00</td>
<td>0.0000000E+00</td>
<td>2.0000000E+00</td>
<td>2.0000000E+01</td>
<td>1.0400000E+00</td>
<td>0.0000000E+00</td>
</tr>
<tr>
<td>302</td>
<td>-5.8637600E-04</td>
<td>0.0000000E+00</td>
<td>0.0000000E+00</td>
<td>0.0000000E+00</td>
<td>0.0000000E+00</td>
<td>4.691088E-03</td>
</tr>
<tr>
<td>303</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
The above OLOAD RESULTANT is from run inrel2.dat and includes the output for the three applied loads (subcases 101, 201, and 301), followed by the output for the six unit accelerations. Note that the subcases are properly identified by the subcase ID, while the loads resulting from unit accelerations are identified by integer numbers beginning with a number one larger than the largest subcase ID. Once again, this output will be printed for each superelement in the model, and the resultants are the summation of all loads applied to interior elements and grid points for superelements defined in the main bulk data. (For PARTs, they are the summation of all loadings defined within the PART’s bulk data.)

The final six loadings are the resultants of the unit accelerations applied to the physical mass of the current superelement (without including any upstream mass). These loadings do not include any concentrated mass elements attached to external grid points for superelements defined in the main bulk data, but these loadings do include any mass defined within the partitioned bulk data for PARTs.

The OLOAD RESULTANT is printed for each superelement as the superelement is processed. At the residual structure, the program calculates the inertial loadings required to put the structure into a state of static equilibrium.

At the residual structure, the program also solves for the matrix DM, each column of which represents the static solution for the model when one SUPORT DOF is moved one unit, while the other SUPORT DOFs are constrained. This matrix is not printed. Rather, the strain energy resulting from that solution is printed in a table preceded by User Information Message 4158 for datablock KLL.

*** USER INFORMATION MESSAGE 4158 (DFMSA)  
-----STATISTICS FOR SPARSE DECOMPOSITION OF DATA BLOCK KLL FOLLOW  
NUMBER OF NEGATIVE TERMS ON FACTOR DIAGONAL - 0  
MAXIMUM RATIO OF MATRIX DIAGONAL TO FACTOR DIAGONAL = 3.0E+03 AT ROW NUMBER 38  
0*** USER INFORMATION MESSAGE 3035 FOR DATA BLOCK KLR  
ENERGY EPSILONS LARGER THAN .001 ARE FLAGGED WITH ASTERISKS  
SUPPORT PT. NO. EPSILON STRAIN  
1 1.4363500E-13 7.3050614E-09  
2 1.4363500E-13 2.5611371E-09  
3 1.4363500E-13 2.5465852E-11  
4 1.4363500E-13 1.4478019E-09  
5 1.4363500E-13 2.4965630E-10  
6 1.4363500E-13 1.3863602E-07  

The values printed in the above table should be numeric zero (that is, essentially zero). The program does not check for this requirement, because NX Nastran is non-dimensional. A value that represents numeric zero when using one set of units may represent a finite value for another set of units. When using inertia relief, always review this table. The SUPPORT PT. NO. is the DOF ID for the reference set (r-set), which is defined by the SUPORT or SUPORT1 entry. If any of the strain energy values is not numeric zero, or if the EPSILON value is not numeric zero, then the model is overconstrained (not capable of rigid body motion in the associated direction) or has other problems.

After this output, several matrices are printed. The matrix QRR represents the rigid body mass matrix of the total structure, including all superelements. Calculated at the reference grid point, the matrix QRR is presented as a 6 x 6 matrix, using global coordinates. Masses on scalar points are ignored in this calculation.

**Intermediate Matrix ... QRR**  
COLUMN 1  
1.583215E-03 -9.274533E-14 0.000000E+00 0.000000E+00 0.000000E+00 -5.204673E-03  
COLUMN 2  
1.583215E-03 0.000000E+00 0.000000E+00 0.000000E+00 6.332861E-04  
COLUMN 3  
0.000000E+00 0.000000E+00 1.583215E-03 5.204673E-03 -6.332861E-04 0.000000E+00
The matrix QRL is the resultant of the applied loads, measured at the reference point. There is one column for each loading condition.

The matrix URACCEL (not produced in this example) is the user-supplied rigid body acceleration, which is input on the DMG,UACCEL entry. This output does not appear if the entry is not present.

The matrix URA represents the rigid body acceleration matrix, computed from the applied loads. Once these accelerations are computed, inertial loads are computed that will place the structure into a state of static equilibrium.

Once the inertial loads required to produce a state of static equilibrium are calculated, these inertial loads are applied and combined with the applied loads. The program then constrains the SUPORT (reference) DOFs and performs a static solution. The output from this point on is conventional superelement output.
Chapter 6: Multiple Loadings in Static Analysis

- Introduction
- How Case Control is Internally Partitioned and Used
- Examples for Multiple Loading Conditions
6.1 Introduction

Although “Loads, Constraints, and Case Control in Static Analysis” discussed multiple loadings, this chapter clarifies how the case control is handled in static analysis with multiple loading conditions.

If multiple load cases exist in a static solution, a separate subcase must exist for the residual structure for each load condition or boundary condition. Separate subcases can exist for the superelements for each solution. Once again, we recommend that either SUPER=ALL (the default) or separate subcases for each superelement for each solution should exist. This section describes how the program internally partitions the case control into separate sections for each superelement.

6.2 How Case Control is Internally Partitioned and Used

When NX Nastran is processing your input file, the bulk data is divided into separate sections (internal tables) based on your superelement definitions. In addition, the case control is also partitioned into separate sets for each superelement. (The name of this separate datablock is CASES in the DMAP.) You have already seen how the SUPER command is used to determine which SUBCASEs apply to which superelements. This command is also used in partitioning the case control.

Based on your use of the SUPER command, a separate set of case control is created for each superelement. One set is for the residual structure. You could think of the residual set as the master set of case control, which is controlling the actual solution. Once again, a SUBCASE must be defined for the residual structure for each loading condition (or boundary condition) you wish to solve. As mentioned before, the program actually copies any SUBCASEs that reference superelement 0 into a CASES table, which is qualified with SEID = 0. When processing the residual structure (PHASE 1), the program uses this copy of the case control.

The program creates a separate CASES table for each superelement in the model. The SEP2CT module creates this table during execution of the PHASE0 subDMAP in the program.

The following example shows (in an idealized form) the case control partitioning.

6.3 Examples for Multiple Loading Conditions

The following is a copy of the case control from Figure 4-7. In this set of case control, we are considering three separate loading conditions. On the next few pages, we show the idealized partitioned case control for each superelement in the model.

```
TITLE = S.E. SAMPLE PROBLEM 1
SUBTITLE = S.E. STATICS - MULTIPLE LOADS
DISP = ALL $ DEFAULT FOR ALL SUBCASES
$ $ SET DEFAULT VALUE FOR SUBCASES
$ SET 999 = 0,3,4,5,6,7
SUPER = 999
$
SUBCASE 101
SUPER = 1,1 $ SUPERELEMENT 1 - RESIDUAL LOAD 1
LABEL = S.E. 1 PRESSURE LOAD
LOAD = 101
$
SUBCASE 102
LABEL = S.E. 2 PRESSURE LOAD
SUPER = 2,1 $ SUPERELEMENT 2 - RESIDUAL LOAD 1
```
LOAD = 102
ELFORCE = ALL $ GET ELEMENT FORCES FOR S.E. 2 - R.S. LOAD 1

SUBCASE 201
SUPER = 1,2 $ SUPERELEMENT 1 - RESIDUAL LOAD 2
LABEL = S.E. 1 - 2# NORMAL LOAD
LOAD = 201

SUBCASE 202
LABEL = S.E. 2 - 2# NORMAL LOAD
SUPER = 2,2 $ SUPERELEMENT 2 - RESIDUAL LOAD 2
LOAD = 202

SUBCASE 301
LABEL = S.E. 1 - OPPOSING LOADS
SUPER = 1,3 $ SUPERELEMENT 1 - RESIDUAL LOAD 3
LOAD = 301

SUBCASE 302
LABEL = S.E. 2 - OPPOSING LOADS
SUPER = 2,3 $ SUPERELEMENT 2 - RESIDUAL LOAD 3
LOAD = 302

SUBCASE 1001 $ USE DEFAULT VALUE FOR SUPER
LABEL = SE 0,3,4,5,6,7 LOAD 1 - PRESSURE
SPCFORCES = ALL

SUBCASE 1002
LABEL = SE 0,3,4,5,6,7 LOAD 2 - NORMAL FORCE

SUBCASE 1003
LABEL = SE 0,3,4,5,6,7 LOAD 3 - OPPOSING LOADS

Idealized form of the CASES table (CASE CONTROL) for the residual structure (SE 0). —The copies for superelements 3, 4, 5, 6, and 7 are identical. SUBCASEs 1001, 1002, and 1003 do not contain a SUPER command, so they get the default value of SUPER = 0,3,4,5,6,7 from above the SUBCASEs.

TITLE = S.E. SAMPLE PROBLEM 1
SUBTITLE = S.E. STATICS - MULTIPLE LOADS
DISP = ALL $ DEFAULT FOR ALL SUBCASES

SUBCASE 1001 $ USE DEFAULT VALUE FOR SUPER
LABEL = SE 0,3,4,5,6,7 LOAD 1 - PRESSURE
SPCFORCES = ALL

SUBCASE 1002
LABEL = SE 0,3,4,5,6,7 LOAD 2 - NORMAL FORCE

SUBCASE 1003
LABEL = SE 0,3,4,5,6,7 LOAD 3 - OPPOSING LOADS

Idealized form of the CASES table (CASE CONTROL) for superelement 1. —SUBCASEs 101, 201, and 301 contain SUPER commands pointing to superelement 1, so they are placed in the CASES table for this superelement.

TITLE = S.E. SAMPLE PROBLEM 1
SUBTITLE = S.E. STATICS - MULTIPLE LOADS
DISP = ALL $ DEFAULT FOR ALL SUBCASES

SUBCASE 101
SUPER = 1,1 $ SUPERELEMENT 1 - RESIDUAL LOAD 1
LABEL = S.E. 1 PRESSURE LOAD
Idealized form of the CASES table (CASE CONTROL) for superelement 2. —SUBCASEs 102, 202, and 302 contain SUPER commands pointing to superelement 2, so they are placed in the CASES table for this superelement.

When performing the solution to this problem, the program combines the first SUBCASEs for each superelement to get the first solution, the second SUBCASEs for each superelement to get the second solution, and so on, for as many solutions as are requested.

Therefore, the first solution is a combination of SUBCASE 101 (applied on superelement 1), SUBCASE 102 (applied on superelement 2), and SUBCASE 1001 (applied on superelements 3, 4, 5, 6, 7, and the residual structure).

These examples demonstrate how the program combines the superelements and obtains the desired solutions. When we get to the section on dynamic analysis, this section will be a valuable reference.
Chapter 7: Multilevel Superelement Analysis

- The Concept of Multilevel Analysis
- Comparison of Single- and Multilevel Analysis
- User Interface for Multilevel Superelements
- Example: Multilevel Problem Solved by Hand
- Examples of Multilevel Superelements
7.1 The Concept of Multilevel Analysis

Multilevel superelement analysis occurs when one or more points that are exterior points of any superelement are interior to another superelement. If this is the case, then a single-level tree is no longer possible. That is, the information resulting from one (or more) superelement(s) is necessary to process another superelement.

In a single-level analysis, all superelements can be processed independently of each other; however, all superelements have to be processed before the residual structure can be processed. In multilevel analysis, that requirement still exists, but it is possible to create assemblies of superelements. That is, the exterior points of several superelements can be interior to a downstream collector superelement that then creates an assembly.

Single-level analysis is the default approach if PARTs are used. You can only use multilevel analysis in a model that contains PARTs by either using the SETREE Bulk Data entry or the DTI,SETREE Bulk Data entry. See “User Interface for Multilevel Superelements” for a description of these entries and how they are used.

If the model consists of main bulk data superelements only, the program checks to see that all exterior points belong to the residual structure before performing a single-level analysis. If a model is defined using the main bulk data only and exterior points exist that do not belong to the residual structure, then the program automatically performs a multilevel analysis. The SETREE and DTI,SETREE entries can also be used, in this case, to control the processing order and connectivity of the superelements.

7.2 Comparison of Single- and Multilevel Analysis

Single-Level Analysis

Up to this point, this guide has described only the case of single-level superelement analysis (that is, all exterior points of any superelements belong to the residual structure). In this case, the reduced matrices of each superelement connect only to points that belong to the residual structure, and each superelement can be processed independently of all others. However, all superelements must be processed before the residual structure can be processed.

Advantages

Single-level processing is the simplest way to define superelements and is recommended for your initial use of superelements. As you become an experienced user of superelements, you can try using multilevel analysis for efficiency (or for another reason, such as multi-step dynamic reduction).

In dynamic analysis, single-level processing is ideal for lightly coupled structures. That is, if there is very little dynamic coupling between components (for example, the interface connection is rigid), then a single-level solution is fine.

During the initial phases of a project, when any part of a structure can be changed, restarts with model changes may be cheaper than using multilevel superelements.

Single-level analysis is the default if PARTs are used.
Sample of a Single-Level Processing Tree

The aircraft model shown in Figure 7-1 is divided into six superelements. The interface points between each superelement are placed in the residual structure. Because each superelement connects only to points that belong to the residual structure, we obtain a single-level processing tree, as shown in Figure 7-2.

![Aircraft Model](image)

**Figure 7-1. Aircraft Model**

![Single Processing Level Tree](image)

**Figure 7-2. Single Processing Level Tree**

Because all the interface points between superelements belong to the residual structure, this single-level tree has a large residual structure, compared to that of a multilevel tree.

We should point out that superelements 5 and 6 are disjoint superelements. A disjoint superelement consists of more than one unique piece, with no direct connection between the pieces (at least inside the current superelement). Superelement 6 consists of the two wing tips of the plane, and superelement 5 consists of the two inboard wing pieces. (We know that in a real aircraft the wing structure is continuous through the fuselage, but for purposes of visualization the model is broken up in this manner.)
The following figures demonstrate the idea of disjoint superelements. In each figure, an x is used to indicate the exterior points (in this case residual structure points to which superelement 6 attaches). In this simplified representation, only a few exterior points are shown on each edge. In a real aircraft model, there might be hundreds of grid points on these boundaries.

![Figure 7-3. Wing Tip Superelement](image)

**Figure 7-3. Wing Tip Superelement**

Figure 7-3 shows the model of the outboard wing tip superelement (SEID = 6). This model contains the interior points, exterior points, and elements for superelement 6. As NX Nastran processes superelement 6, it first creates a set of G-sized matrices (including both the interior and exterior points) and then applies (MPC- and SPC-type) constraints. These steps are then followed by a reduction process to A-sized matrices representing the mass, damping, loading, and stiffness of superelement 6, as seen by the residual structure. Graphically, we can idealize these matrices as follows (the x’s represent the residual points to which superelements connect):

![Figure 7-4. Reduction Process, SEID=6](image)

**Figure 7-4. Reduction Process, SEID=6**

Superelement 5 would be treated in a similar manner. Figure 7-4 shows the physical model for superelement 5.
Once again, NX Nastran generates a set of G-sized matrices and reduces them to the exterior DOFs. This process is idealized in the following representation of the reduced matrices for superelement 5. Once again, in these figures, the x’s represent the residual structure points to which superelement 5 connects.

All other superelements are processed, and at this point we can show an idealized representation of the residual structure for the single-level tree version of the airplane model.
Figure 7-7. Reduction Process, Total Model

Although this model no longer resembles the physical airplane model with which we started, it provides the same answers as the full finite element model for static analysis.

In this idealized single-level representation, the x’s represent the residual structure points and the circled numbers represent the reduced matrices for the superelements. The lines indicate the residual structure points to which each superelement connects. Because this is a single-level model, connection (attachment) between superelements does not occur until the residual structure, when all of the pieces are assembled. In a single-level tree, any points where a connection between superelements occurs is interior to the residual structure.

The advantages of single-level analysis are most obvious during the initial phases of a project, when any part of a design can change. In a single-level analysis, only the superelements containing the changes need to be reprocessed; the rest of the original superelement matrices are assembled with these new matrices at the residual structure, to create the new solution. Although only small parts of the model needed to be processed, correct answers are available for the entire model, including deformed plots and stress contours in your postprocessor.

Multilevel Analysis

In multilevel superelement analysis, superelements can connect to the residual structure and directly to each other. Or more simply, a point that is exterior to a superelement can be interior to another superelement or to the residual structure. In this case, a predefined processing order is necessary before processing the superelements, because one or more superelements might need to be processed before another (assembly) superelement.

You can create assemblies with multilevel processing, that is, superelements that are actually an assembly of one or more superelements. For example, we will look at the wings of the aircraft model used to show the single-level tree. Let us assume that we are not expecting to make changes in the wing area (although we are able to make changes should we need to), so we may want to create a new wing assembly superelement. By removing grid points from the residual structure, we can reduce its size to make the final solution and model change restarts more efficient.
Sample of Multilevel Analysis, Example 1

For this case, we create a new collector superelement, which we will call superelement 56. (We chose this number to imply that we have combined superelements 5 and 6 to create this assembly—any unused number could have been used.) We create this new collector by placing the grid points that lie on the boundary between superelements 5 and 6 into the new superelement (56). It is also advisable (mandatory, if PARTs are used) to define the processing tree—a topic that is covered later in this section.

Figure 7-8 shows the original (single-level) model for the wings containing superelements 5 and 6 and the grid points connecting them.

![Figure 7-8. Wing Model Single Level SE](image)

To create a collector superelement, we put the grid points lying on the interface between these superelements into superelement 56. (If PARTs are used, you can also define collector superelement 56 in the processing tree.) This new definition of superelements is shown in Figure 7-9.

![Figure 7-9. Wing Model, Multilevel Collector SEID=56](image)

This new definition results in a modified processing tree, shown in Figure 7-10.
Chapter 7: Multilevel Superelement Analysis

Figure 7-10. Multilevel Tree with Collector SEID=56

The points where the wings connect to the fuselage still belong to the residual structure, but we now have a collector superelement for the wing assembly. Superelements 5 and 6 are processed exactly as before. That is, when these superelements are processed, the points on the interface between them and the points on the wing-to-fuselage interface are still exterior. We only see the differences after superelements 5 and 6 have been processed.

Figure 7-11, which shows superelement 6 in this new configuration, is almost identical to Figure 7-3; the only difference is that the grid points on the interface between the wing components now belong to superelement 56.

Figure 7-11. Wing Tip with Collector

After superelement 6 is processed, we have the same reduced matrices as before, shown in Figure 7-12. The matrices connect to the same grid points as before; however, these grid points now belong to superelement 56 instead of the residual structure.
Superelement 5 is similar to what it was before, as shown in Figure 7-13. The only difference is that the grid points on the interface between the wing components belong to superelement 56, rather than to the residual structure.

Once again, after processing we have the same matrices as before; however, the matrices now connect to superelements 0 and 56, rather than to the residual structure only.

At this point, we can process superelement 56. This superelement is a collector superelement, which combines the reduced matrices from superelements 5 and 6 into an assembly and then reduces those matrices to the wing-to-fuselage interface points. Figure 7-15 shows the physical model for superelement 56. Because all the elements for the wings have been placed into superelements 5 and 6, the physical model for superelement 56 has only the grid points we have assigned to it.
Note

A collector—or assembly—superelement can have elements in it; however, this example does not.

Figure 7-15. Collector SEID=56

If there were any elements in the physical model for superelement 56, we would create the physical mass and stiffness of this component and then add the reduced mass and stiffness from the upstream superelements to get the assembly matrices. Because there are no elements in this superelement, NX Nastran does not create a physical mass or stiffness, only the assembly matrices. It treats the upstream superelements as if they were finite elements. That is, the program uses the superelement tree to determine which superelements must be contained within superelement 56 (superelements 5 and 6) and then adds these superelements to the physical model of superelement 56. The complete model of superelement 56 includes any physical elements in the superelement, plus the reduced matrices from the upstream superelements.

Now we build the assembly model. First we have the physical model of superelement 56 (Figure 7-15). Then we add the reduced matrices for superelements 5 and 6 to get the assembly. We will show this process step-by-step for clarity.

First we attach superelement 6 to the physical model, resulting in the model shown in Figure 7-16. (Note that this model looks like the reduced model for superelement 6, because the interior points for superelement 56 are the exterior points for superelement 6.)

Figure 7-16. Physical Model, SEID=56

We are now ready to add superelement 5 to the assembly. Superelement 5 has as its exterior points not only the interior points of superelement 56, but also the grid points that lie on the interface between the wings and the fuselage. Therefore, the model of superelement 56 has these points as exterior points, as shown in the assembly in Figure 7-17.
Figure 7-17. Representation of the Assembly Model for Superelement 56.

In Figure 7-17, interior points are shown by dots, and exterior points are shown by x. At this point, G-sized mass, stiffness, loading, and damping matrices are created, and the collector superelement goes through the standard reduction process. The reduced model can be idealized, as shown in Figure 7-18.

Figure 7-18. Idealized Representation of Reduced Collector Superelement 56.

From this point on, the processing is the same as before. Figure 7-19 shows the idealized residual structure for this model.

Figure 7-19. Residual Structure Model.

An advantage of this method is the reduction in size of the residual structure, which is accomplished by moving points upstream into one or more collector superelements. Also, if a restart run has no
changes for superelements 5, 6, or 56, that branch of the tree is not reprocessed and the restart is more efficient than a single-level tree, due to the reduced size of the residual structure problem.

A Second Multilevel Tree Example

Multilevel analysis can also be used to create assembly superelements by placing one or more superelements above another superelement in the tree. If we are using PARTs, we can create assembly superelements by providing the multilevel tree. If no PARTs are present, a multilevel tree can be accomplished by placing any grid points that lie on the boundary between two superelements into one of the superelements (although we recommend also defining the processing tree). Figure 7-20 shows the multi-level processing tree we are using now.

![Multilevel Tree Number 2, Showing Assemblies](image)

**Figure 7-20.** Multilevel Tree Number 2, Showing Assemblies.

To create this processing tree, we have placed grid points that once belonged to the residual structure interior to superelements. For example, the grid points where superelements 5 and 6 meet have been placed interior to superelement 5, reducing the size of the residual structure. Figure 7-21 shows how the boundary grid points have been placed into superelements to create this multilevel tree.

![Superelement Definition for Multilevel Tree 2](image)

**Figure 7-21.** Superelement Definition for Multilevel Tree 2.
This is a good processing tree for dynamic analysis. Although we have not discussed dynamic analysis using superelements, it is evident that this model has a processing tree that resembles the way the actual structure is connected. That is, if we look at the wings, we see that superelement 6 is cantilevered off superelement 5. Looking at the fuselage and tail assembly, we see that the tail (superelement 4) cantilevers off the rear fuselage (superelement 3), and the rear fuselage cantilevers off the mid-fuselage (superelement 2).

In the sections discussing superelements in dynamic analysis, we see that this type of tree can be an ideal means to provide maximum accuracy with the minimum residual structure. For example, think of the wings—the wing tips (superelement 6) cantilever off the inner wings (superelement 5). If we were to calculate the modes of the wing tips with the interface between them and the inner wings constrained, those modes would look very similar to the way we would expect the wing tips to work in the assembly.

In the wing assembly (superelement 5), we see that if we attach the wing tips to the inner wings and then calculate cantilevered modes of that assembly, those modes also look similar to the way we would expect the wings to behave if attached to the fuselage. Thus, multilevel analysis can greatly improve the results in a dynamic analysis, while reducing the size of the final solution (the residual structure).

Looking at the wing assembly again, we go through the reduction process. First we start at the top of that branch of the tree; superelement 6 is the first superelement on that branch to be processed. Figure 7-22 shows that superelement 6 looks exactly as it did in the previous examples; however, the exterior points belong to superelement 5, rather than the residual structure or superelement 56.

![Figure 7-22. Superelement 6 for the Second Multilevel Tree.](image)

Once again, superelement 6 is processed and reduced to matrices connected to the exterior points. The physical model is replaced by matrices as idealized in Figure 7-23.
Now that superelement 6 is processed, we are ready to proceed to the next lower superelement in this branch—superelement 5. Superelement 5 is no longer simply the inner wing, rather it is the complete wing assembly, idealized as follows. First we create the physical model of superelement 5.

This physical model of superelement 5 contains the interface grid points between the inner wing and the wing tips as interior points. The only exterior points are the connections between the wings and the fuselage.

Now we look at the processing tree and see that superelement 6 is upstream from superelement 5. Therefore, we add the reduced matrices for superelement 6 to the model for superelement 5, creating the wing assembly superelement.
Superelement 5 is processed and replaced by a set of matrices that represent the properties of the wing assembly, as seen by the connection to the fuselage.

The tail-fuselage assembly is created in a similar manner. After all superelements are processed, the grid points at the connection between the fuselage and the wings are all that remain in the residual structure. These points are connected by matrices representing assembly superelements 2 and 5.

As you can see, the residual structure has been reduced significantly from the single-level configuration. If you want multiple solutions, the multilevel method is much more efficient.

**A Third Example of a Multilevel Tree**

The processing in multilevel analysis does not need to be physically intuitive; that is, the processing order can be varied to satisfy particular interests or needs. For example, we assume that the aircraft is far along in the design process (or perhaps we are updating an existing plane), and the only areas of the model we expect to change are the tail, middle fuselage, or forward fuselage. The following processing tree (Figure 7-28) allows us to use a smaller residual structure than the single-level tree, while being efficient for changes in superelements 1, 2, or 4. This fact does not mean that a change in superelements 3, 5, or 6 would be inefficient; however, the tree is optimized for changes to
superelements 1, 2, or 4. Note that superelement 3 is above superelement 4 in the tree. Though we would not normally process these superelements in this way, we have placed superelement 4 immediately above the residual structure so that applying changes to this superelement is efficient.

![Multilevel Tree for the Third Multilevel Example](image)

**Figure 7-28. Multilevel Tree for the Third Multilevel Example.**

Figure 7-29 shows how the interface points have been moved into superelements to create this multilevel tree.

![Model for the Third Multilevel Example](image)

**Figure 7-29. Model for the Third Multilevel Example.**
For this example, we look at the tail and aft fuselage superelements. First, starting at the tip of that assembly, we process the aft fuselage. The physical model of this superelement is shown in Figure 7-30, with the exterior points belonging to superelements 4 and 0.

![Figure 7-30. Model of Superelement 3 for the Third Multilevel Example.](image)

This superelement is processed and replaced by reduced matrices, as idealized in Figure 7-31.

![Figure 7-31. Model for the Third Multilevel Example.](image)

We are now ready to proceed to superelement 4 to create the tail-aft fuselage assembly. First we create the physical model of superelement 4, as shown in Figure 7-32.

![Figure 7-32. Physical Model of Superelement 4.](image)

In the physical model of superelement 4, no exterior points have been identified. It is only after we add the reduced model of superelement 3 that we see the connection to exterior points belonging to the residual structure. We now add the reduced model of the upstream superelement (superelement 3) to this model and obtain the model of assembly superelement 4.
Figure 7-33. Complete Model of Assembly Superelement 4.

This superelement is processed and replaced by reduced matrices, which represent the properties of the aft assembly, as seen by the rest of the model.

Figure 7-34. Idealized Representation of the Aft Assembly.

After the other branches of the tree have been processed, the complete assembly is created and solved at the residual structure.

As you can see, multilevel superelements give you complete control over the solution process. Single-level superelement analysis is simpler, but it requires all superelements to connect only to the residual structure. With multilevel analysis, you can create assemblies and reduce the size of the residual structure, resulting in increased efficiency for model changes.

7.3 User Interface for Multilevel Superelements

There are several ways to define multilevel superelement processing. These definitions are divided into two sections—models that are main bulk data only and models containing PARTs.

This section deals with models for static analysis. There are additional options in dynamic analysis, and the section for using superelements in dynamic analysis provides additional information.

Multilevel Processing When the Model is Main Bulk Data Only

Multilevel processing can be defined automatically or manually when the model is main bulk data only. The automatic approach is used if any point that is exterior to a superelement is interior to another superelement. The manual approach is controlled by the DTI,SETREE Bulk Data entry. We recommend the manual approach. (If the model does not contain PARTS, the SETREE entry will not work.)

Automatic Creation of the Processing Tree

If the model is main bulk data only, and there is no DTI,SETREE entry in the bulk data, NX Nastran attempts to automatically determine a processing tree that satisfies the requirements of the model provided. There are limitations on whether NX Nastran can create a processing tree. These limits are:
• An element can connect to no more than two superelements and the residual structure.

• Each superelement can connect to only one downstream superelement (no connections across branches of the processing tree).

The first limitation is internal to the routine that tries to find the processing tree; the second restriction is a program limitation. Do not misinterpret the second limitation to mean that exterior points of a superelement must belong to a single downstream superelement, that is, that a superelement can connect only to one downstream superelement in the processing tree. For example, in the third sample of the multilevel tree for the aircraft shown earlier, superelement 3 has exterior points that are interior to both superelement 4 and the residual structure. Because superelement 4 is downstream from superelement 3, this approach works. The grid points at the interface between the aft fuselage and the mid-fuselage are not only exterior to superelement 3, they are also exterior to superelement 4.

We do not recommend having NX Nastran determine a multilevel processing tree automatically. Although there is nothing wrong with allowing this choice, the program does not know the physical structure with which you are working or the areas of the model you are planning to change. Because you are knowledgeable about these items, it is more efficient for you to select the processing tree. Also, on automatic restart you can make changes in the model that would cause the program’s automatic restart logic to change the processing tree, thus increasing the computation required to get the answer. Therefore, you should always provide the processing tree for multilevel problems.

There is a Case Control command that can be used to assist the automatic logic. This command is SEFINAL. The SEFINAL command forces the automatic process to place one or more superelements at the level immediately above the residual structure in the processing tree. If you use the automatic creation of the processing tree, you may want to use this command to select the superelement (or set of superelements) that will be at the lowest level above the residual. In this way, you can attempt to improve the restart efficiency when you are making changes.

**Manual Definition of the Processing Tree for a Model Defined in the Main Bulk Data Section Only**

There are two main bulk data entries that can be used to define the processing tree. One entry is DTI,SETREE. This entry can be used to manually define a multilevel processing tree for a model defined in the Main Bulk Data Section only. If the model contains PARTs, either the DTI,SETREE or SETREE entries are mandatory to obtain a multilevel tree.

The DTI, SETREE used to be the only Bulk Data entry that could be used to define the multilevel tree. The SETREE entry now offers an alternative method.

With the DTI,SETREE entry, you can specify the entire processing tree using one entry.

The DTI,SETREE is a table that you provide. The table is provided as record 1 (field 3 on the entry). After the record number, pairs of superelements are provided [seupi-sedowni] until the tree is defined. Continuation lines can be used.

<table>
<thead>
<tr>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
</tr>
</thead>
<tbody>
<tr>
<td>DTI</td>
<td>SETREE</td>
<td>“1”</td>
<td>SEUP1</td>
<td>SEDOWN1</td>
<td>SEUP2</td>
<td>SEDOWN2</td>
<td>SEUP3</td>
<td>SEDOWN3</td>
<td></td>
</tr>
<tr>
<td></td>
<td>SEUP4</td>
<td>SEDOWN4</td>
<td>SEUP5</td>
<td>SEDOWN5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>etc.-</td>
</tr>
</tbody>
</table>

The following example defines the tree shown in Figure 7-28.

DTI,SETREE,1,1,0,6,5,5,2,
Notice that the superelements are listed in upstream and downstream pairs, and the entire tree is given using one entry.

**Multilevel Processing When the Model Uses PARTs**

If the model includes PARTs, you must use the SETREE or DTI,SETREE entries if you want a multilevel tree is. Models containing PARTs are automatically treated as single-level, unless one of these entries appears in the Main Bulk Data Section. Otherwise, the SETREE and DTI,SETREE define a tree in the same manner as models defined in the Main Bulk Data Section only. Once again, these entries must be in the Main Bulk Data Section of the input file.

The SETREE entry actually provides a list of all superelements immediately upstream from a single superelement in the tree. One SETREE is required for each superelement that has any upstream connections.

<table>
<thead>
<tr>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
</tr>
</thead>
<tbody>
<tr>
<td>SETREE</td>
<td>SEID</td>
<td>SEUP1</td>
<td>SEUP2</td>
<td>SEUP3</td>
<td>SEUP4</td>
<td>SEUP5</td>
<td>SEUP6</td>
<td>SEUP7</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>SEUP8</td>
<td>SEUP9</td>
<td>-etc.-</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

The following set of SETREE entries accomplishes the same tree as the above DTI,SETREE for Figure 7-28.

- SETREE, 0, 1, 2, 4
- SETREE, 2, 5
- SETREE, 5, 6
- SETREE, 4, 3

In this set, the first entry states that superelements 1, 2, and 4 are directly above the residual structure. The second entry states that superelement 5 is directly above superelement 2, and so on.

Either the SETREE or the DTI,SETREE can be used, and if both entries are in the input file, the SETREE takes precedence. Also note that these entries must be in the Main Bulk Data Section if they are used. Verify that all superelements you want to include in the analysis are listed on the SETREE (or DTI,SETREE) entries, because superelements that are not listed are placed directly above the residual structure.

### 7.4 Example: Multilevel Problem Solved by Hand

An example solved by hand is the best demonstration of how multilevel superelements are processed. Because solving large problems by hand is not practical, only the simplest of problems can be used. In this case, we use a fixed-fixed beam that is divided into multilevel superelements. We only look at motion in the horizontal direction in the plane of the paper. The model is shown below.
For this example, we use the following SESET and DTI,SETREE entries.

SESET,1,6,7
SESET,2,4,5
DTI,SETREE,1,2,0,1,2

Using this superelement definition, the following model for superelement 1 is created. The program removes grid points 6 and 7 (interior points) from the bulk data. The program then removes all elements connected to those points and makes a copy of grid point 5, which is the exterior point for this superelement. Any loads and constraints associated with grid points 5 and 6 or the elements in superelement 1 are also placed in the bulk data for the model.

The first step in processing the superelement is to generate the G-sized matrices for this component. When dealing with a multilevel configuration, there are two sets of matrices generated for each superelement.

The first set of matrices created represents the physical properties of the superelement (plus any K2GG, K42GG, M2GG, B2GG, or P2G matrices). The first set of matrices are G-sized and are defined by the letter J. For example, the physical stiffness matrix is called KJJ. The following stiffness and loading matrices are generated for superelement 1.
Chapter 7: Multilevel Superelement Analysis

\[
[K_{JJ}] = \begin{bmatrix} 1. & -1. & 0. \\ -1. & 2. & -1. \\ 0. & -1. & 1. \end{bmatrix} \quad \text{in term of} \quad \begin{bmatrix} U_5 \\ U_6 \\ U_7 \end{bmatrix}
\]

\[
[P_J] = \begin{bmatrix} 0. \\ 4. \\ 0. \end{bmatrix}
\]

Equation 7-1.

The program then creates assembly G-sized matrices by combining these J matrices with the reduced matrices from upstream superelements. Because superelement 1 has no upstream superelements, the J and G matrices are the same. (Note that for this situation, where two matrices are identical, NX Nastran stores only one matrix in the database and creates a pointer for the second matrix, which points to the stored data. Therefore, the database files do not become excessively large.)

If there were any MPC-type relations (MPCs, RBE2, RBAR, etc), the program would apply and process these relations, and the G matrices would be reduced to N matrices.

The program then applies constraints to the matrices. For superelement 1, grid point 7 is constrained; thus, terms associated with this grid point are removed from the matrices to apply that constraint. After applying the constraints, the matrices are defined for the Fset.

\[
[K_{FF}] = \begin{bmatrix} 1.0 & -1. \\ -1. & 2.0 \end{bmatrix} = \begin{bmatrix} \bar{K}_{AA} & \bar{K}_{AO} \\ \bar{K}_{OA} & \bar{K}_{OO} \end{bmatrix}
\]

\[
[P_F] = \begin{bmatrix} 0. \\ 4. \end{bmatrix} = \begin{bmatrix} P_A \\ P_O \end{bmatrix}
\]

Equation 7-2.

At this point, the static condensation is performed. The matrices are partitioned into A and O set DOFs and then transformed.

\[
[G_{OA}] = -[K_{OO}]^{-1}[K_{OA}] = -[.5] \cdot [-1.] = [.5]
\]

\[
[K_{AA}] = [K_{AA}] + [K_{OA}]^T[G_{OA}] = (1.) - (.5) = [.5]
\]

\[
[P_A] = [P_A] + [G_{OA}]^T[P_O] = [2.0]
\]

Equation 7-3.
Superelement 1 is now processed, and only the set of \( A \) sized matrices are required to continue the solution.

**SUPERELEMENT 2**

The physical model for superelement 2 is shown below. This model contains interior points 4 and 5, any remaining elements that are connected to them, a copy of exterior point 3, and the reduced matrices representing superelement 1.

![Physical model of superelement 2](image)

Once again, the \( J \) matrices are generated for the physical model of the superelement.

\[
[K_{JJ}] = \begin{bmatrix}
1.0 & -1.0 & 0.0 \\
-1.0 & 2.0 & -1.0 \\
0.0 & -1.0 & 1.0
\end{bmatrix}, \quad [P_J] = \begin{bmatrix}
0.0 \\
0.0 \\
3.0
\end{bmatrix}
\]

Equation 7-4.

Now we add the reduced matrices from superelement 1. Superelement 1 is connected to grid point 5, so its reduced matrices are added to the terms for that grid point. The reduced stiffness of .5 units is added to the existing term in that position (1.0) to get the assembly stiffness of 1.5 in that DOF. The reduced load of 2.0 units is added to the physical load on grid 5 (3.0 units), resulting in an assembly load of 5.0 units on that point. The resulting assembly stiffness and loading matrices are shown below.

\[
K_{GG} = \begin{bmatrix}
1.0 & -1.0 & 0.0 \\
-1.0 & 2.0 & -1.0 \\
0.0 & -1.0 & 1.5
\end{bmatrix}, \quad [P_G] = \begin{bmatrix}
0.0 \\
0.0 \\
5.0
\end{bmatrix}
\]

Equation 7-5.

This superelement has no MPCs and constraints applied, so we proceed to the reduction process.
Residual Structure

The residual structure consists of what remains of the model, plus the reduced assembly matrices from superelement 2. Once again, the physical matrices are generated, providing the following:

$$[K_{FF}] = \begin{bmatrix} 1.0 & -1.0 & 0.0 \\ -1.0 & 2.0 & -1.0 \\ 0.0 & -1.0 & 1.5 \end{bmatrix} = \begin{bmatrix} \bar{K}_{AA} & \bar{K}_{AO} \\ \bar{K}_{OA} & \bar{K}_{OO} \end{bmatrix}$$

$$[G_{OA}] = -[K_{OO}]^{-1}[K_{OA}] = -\begin{bmatrix} .75 & .5 \\ .50 & 1.0 \end{bmatrix} \begin{bmatrix} -1.0 \\ 0.0 \end{bmatrix} = \begin{bmatrix} .75 \\ .50 \end{bmatrix}$$

$$[K_{AA}] = [\bar{K}_{AA}] + [K_{OA}]^T[G_{OA}] = [0.25]$$

$$[P_A] = [\bar{P}_A] + [G_{OA}]^T[P_O] = 0.0 + 2.5 = [2.5]$$

Equation 7-6.

$$\begin{bmatrix} 1.0 & -1.0 & 0.0 \\ -1.0 & 2.0 & -1.0 \\ 0.0 & -1.0 & 1.0 \end{bmatrix} [P_J] = \begin{bmatrix} 0.0 \\ 1.0 \\ 2.0 \end{bmatrix} \text{ in terms of } \begin{pmatrix} U_1 \\ U_2 \\ U_3 \end{pmatrix}$$

Equation 7-7.

Now the reduced matrices are added from upstream assembly superelement 2. The reduced stiffness value of 0.25 units is added into the term for DOF 3, resulting in a diagonal term in the stiffness of 1.25 units. The reduced loads are added into the loading vector at the appropriate location, providing a resulting load of 4.5 units (2.5 + 2.0) at DOF 3.
Now the MPC and SPCs are applied, resulting in the constrained stiffness matrix and loading matrix.

\[
[K_{GG}] = \begin{bmatrix} 1.0 & -1. & 0.0 \\ -1. & 2.0 & -1. \\ 0.0 & -1. & 1.25 \end{bmatrix} \quad [P_G] = \begin{bmatrix} 0. \\ 1. \\ 4.5 \end{bmatrix}
\]

**Equation 7-8.**

Because there is no reduction, the F- and A-matrices are identical. Now we solve the resulting equations to get the displacements for the residual structure.

\[
[K_{FF}][U_F] = [P_F] \quad \text{which gives} \quad [U_F] = \begin{bmatrix} 3.833 \\ 6.667 \end{bmatrix}
\]

**Equation 7-9.**

This solution is then expanded to G size, providing the solution vector for the residual structure.

\[
[U_G] = \begin{bmatrix} 0. \\ 3.833 \\ 6.667 \end{bmatrix} = \begin{bmatrix} U_1 \\ U_2 \\ U_3 \end{bmatrix}
\]

**Equation 7-10.**

Now that we have the displacements for the residual structure, we can calculate stresses, reactions, and all standard output quantities.

Next we look at how data recovery is performed for the superelements, so we proceed up the tree. The superelement directly above the residual structure is superelement 2, so we perform data recovery on superelement 2 first.

As you remember from “Introduction and Fundamentals”, the displacements of a superelement are found using the following equation:

\[
U_o = U_o^0 + G_{oal}U_a
\]

**Equation 7-12.**
First, we find the solution for the exterior points-$U_A$ from the solution for the residual structure.

\[
[U_3] = [6.667] = [U_A]
\]

**Equation 7-13.**

Using this value, we calculate the solution due to boundary motion.

\[
[U^a_O] = [G_{OAT}] [U_A] = \begin{bmatrix} .75 \\ .50 \end{bmatrix} [6.667] = \begin{bmatrix} 5.0 \\ 3.333 \end{bmatrix} = \begin{bmatrix} U_{4A} \\ U_{5A} \end{bmatrix}
\]

**Equation 7-14.**

Now we calculate the fixed-boundary solution.

\[
[U^O_O] = [K_{OO}]^{-1} [P_O] = \begin{bmatrix} .75 & .5 \\ .50 & 1.0 \end{bmatrix} \begin{bmatrix} 0. \\ 5. \end{bmatrix} = \begin{bmatrix} 2.5 \\ 5.0 \end{bmatrix}
\]

**Equation 7-15.**

Combining the two, we obtain the solution for displacements of superelement 2.

\[
[U_O] = [U^a_O] + [U^O_O] = \begin{bmatrix} 7.5 \\ 8.333 \end{bmatrix} = \begin{bmatrix} U_4 \\ U_5 \end{bmatrix}
\]

\[
[U_G] = \begin{bmatrix} 6.666 \\ 7.5 \\ 8.333 \end{bmatrix} = \begin{bmatrix} U_3 \\ U_4 \\ U_5 \end{bmatrix}
\]

**Equation 7-16.**

Now standard output quantities can be calculated for superelement 2. At this point, we are ready to progress up the current branch of the tree and to perform data recovery on superelement 1. The first step for superelement 1 is to obtain the solution at the exterior points (in this case DOF 5).

\[
[U_5] = [8.333] = [U_A]
\]

**Equation 7-17.**

Using this result, we calculate the solution due to the boundary motion.
\[
[U^o_O] = [G_{OA}][U_A] = [0.5][8.333] = [4.166]
\]

Equation 7-18.

Finally, we calculate the fixed boundary solution. Then we combine this result with the solution due to the boundary motion to get the displacements for superelement 1.

\[
[U^O_O] = [K_{OO}]^{-1}[P_O] = [0.5][4.0] = [2.0]
\]

\[
[U_O] = [U^O_O] + [U^o_O] = [4.166] + [2.0] = [6.166] = [U_6]
\]

\[
[U_G] = \begin{bmatrix} 8.333 \\ 6.166 \\ 0.0 \end{bmatrix} = \begin{bmatrix} U_5 \\ U_6 \\ U_7 \end{bmatrix}
\]

Equation 7-19.

For comparison, if we solve this problem as a residual structure-only model, we get

\[
[K_{FF}][\mu_F] = [P_F] = \begin{bmatrix} 2. & -1. & 0. & 0. & \mu_2 \\ -1. & 2. & -1. & 0. & \mu_3 \\ 0. & -1. & 2. & -1. & \mu_4 \\ 0. & 0. & -1. & 2. & \mu_5 \\ 0. & 0. & 0. & -1. & \mu_6 \end{bmatrix} = \begin{bmatrix} 1. \\ 2. \\ 0. \\ 3. \\ 4. \end{bmatrix}
\]

Equation 7-20.

which gives

\[
\begin{bmatrix} U_2 \\ U_3 \\ U_4 \\ U_5 \\ U_6 \end{bmatrix} = \begin{bmatrix} 3.833 \\ 6.667 \\ 7.5 \\ 8.333 \\ 6.166 \end{bmatrix}
\]

Equation 7-21.
The two approaches give identical answers, verifying that, for statics, using superelements for the solution does not introduce any approximation.

### 7.5 Examples of Multilevel Superelements

For multilevel examples, we use the two-headed flyswatter model again, but now we set up the model for multilevel processing.

The model is shown again in Figure 7-35 for reference. The superelements are defined almost exactly as for the examples in "How to Define a Superelement", except that we set up the model as a multilevel solution.

![Figure 7-35. Two-Headed Flyswatter-Entire Model.](image)

We start with the tree shown in Figure 7-36.
The following DTI,SETREE entry can be used to define this tree:

```
DTI,SETREE,1,1,3,3,5,5,0,
,7,0,2,4,4,6,6,0
```

This entry, added to the Main Bulk Data Section, is sufficient to define a multilevel processing tree. However, because we are working in a multilevel tree, we can make the solution more efficient by moving boundary points from the residual structure into upstream superelements, where possible. For example, we will look at the interface between superelements 1 and 3.
Figure 7-37. Superelements and Interface Points for Multilevel Sample.

Grid points 35 and 36 lie on the interface between superelements 1 and 3. In the single-level tree, we placed those points into the residual structure. For the multilevel tree, it is better to put these points into superelement 3 (the lowest superelement on the tree to which they connect), instead of the residual structure. In a model defined by main bulk data superelements, SESET entries can be used to place these points into superelement 3. If PARTs are used, copies of these points need to be in the Bulk Data Sections defining superelements 1 and 3, but not in the section defining the residual structure, however it might be preferrable to have copies in the residual structure data.

Note

When using parts, the points in the different sections do not need the same grid point id, just the same geometric location; the automatic attachment will make the connection.

Input File multi1.dat-Multilevel Superelements Using Main Bulk Data Only

$ multi1.dat - multi-level superelements - main bulk data only
$ SOL 101
CEND
TITLE = file multil -multi-level tree - main bulk data only
SUBTITLE = STATICS
DISP = ALL
$
SUBCASE 101
LABEL = PRESSURE LOAD
LOAD = 101
$
SUBCASE 201
LABEL = 2# NORMAL LOADS
LOAD = 201
$
SUBCASE 301
LABEL = OPPOSING LOADS
LOAD = 301
$
include ‘plot.blk’
$
BEGIN BULK
$
setree,3,1
setree,5,3
setree,4,2
setree,6,4
setree,0,5,6,7
$
include ‘seset.mult’
$
INCLUDE ‘model.dat’
$
INCLUDE ‘load1.dat’
$
ENDDATA

Input File seset.mult

$ FILE SESET.MULT
$
SESET,1,33,34,37,38
SESET,1,45,THRU,50
SESET,1,57,THRU,62
SESET,1,69,THRU,74
SESET,1,81,THRU,86
SESET,1,93,THRU,98
$
SESET,2,39,40,43,44
SESET,2,51,THRU,56
SESET,2,63,THRU,68
SESET,2,75,THRU,80
SESET,2,87,THRU,92
SESET,2,99,THRU,104
$
$ ADD BOUNDARY GRIDS TO DOWNSTREAM S.E.
$
SESET,3,35,36
SESET,3,29,30
$
$ ADD BOUNDARY GRIDS TO DOWNSTREAM S.E.
$
SESET,4,41,42
SESET,4,31,32
$
$ ADD BOUNDARY GRIDS TO DOWNSTREAM S.E.
$
SESET,5,19,20
Notice that grid points 35 and 36 are placed interior to superelement 3. The points on the interfaces between the other superelements are treated similarly.

**Input File multi2.dat-Multilevel Superelements Using PARTs**

```plaintext
$ file - multi2.dat

$ all 7 s.e. brought in using begin super
$ duplicate boundary grids id
$ each s.e. contains its own property description with the same id
$ condensed subcase setup
$ multi-level superelement sample
$ $ assign output2='allsep1.op2',unit=12,delete
id allsepl dat $ jml
SOL 101
TIME 15
CEND
TITLE = S.E. SAMPLE PROBLEM 1
SUBTITLE = S.E. STATICS - RUN 1 - MULTIPLE LOADS
DISP = ALL
stress = all
SUBCASE 101
LABEL = PRESSURE LOAD
LOAD = 101
$ SUBCASE 201
LABEL = 2# NORMAL LOADS
LOAD = 201
$ SUBCASE 301
LABEL = OPPOSING LOADS
LOAD = 301
$ include 'plot.blk'
$ BEGIN BULK
setree,3,1
setree,5,3
setree,4,2
setree,6,4
setree,0,5,6,7
$ param,post,-1
$ $ CQUAD4 5 1 13 14 24 23
$ GRIDSET 6
GRID 13 -.4 3.6 0.
```
The differences between multi2.dat and single2.dat are in the Main Bulk Data Section: The SETREE entries are added to define the multilevel tree, and the interface points between the PARTs are not included in the Main Bulk Data Section and are, therefore, not residual structure points.
Chapter 8: Output in Superelement Static Analysis

• Sorted Bulk Data for PARTs
• Boundary Grid Search Output for PARTs
• Superelement Definition Table
8.1 Sorted Bulk Data for PARTs

When superelements are used in static analysis, there are changes and additions to the standard output. First, the bulk data can be listed as a series of sections. If PARTs are used, the output has the sorted bulk data listing for the Main Bulk Data Section, followed by the section for each PART in the order in which they occur in the input file. Each set of output is labeled SORTED BULK DATA ECHO. The sorted data for the PARTs contains the PART number on this line. For example, each page containing sorted bulk data for PART 1 begins with:

```
SORTED BULK DATA ECHO
SUPER = 1
```

The sorted data for the main bulk data from the input file inrel2.dat is:

```
INPUT BULK DATA CARD COUNT = 25
FILE INREL2.DAT - INERTIA RELIEF USING PARTS  April 29, 2003 NX Nastran 4/29/04 PAGE 5
REFERENCE POINT IS POINT 13 IN RESIDUAL

SORTED BULK DATA ECHO

CARD
COUNT  1  2  3  4  5  6  7  8  9 10
1- CQUAD4  5  1  13  14  24  23
2- GRID  13  -.4  3.6  0.
3- GRID  14  .4  3.6  0.
4- GRID  23  -.4  4.4  0.
5- GRID  24  .4  4.4  0.
6- MAT1  1  30.+6  .3  .283
7- PARAM AUTOSPC YES
8- PARAM INREL -1
9- PARAM POST -1
10- PARAM WTMAS .00259
11- PSHELL  1  1  .05  1  1
12- SUPHIT  13  123456
ENDDATA
TOTAL COUNT= 13
```

Any entries occurring in the Main Bulk Data Section (that is, between BEGIN BULK and the first BEGIN SUPER or the ENDDATA, if no BEGIN SUPER is used) appear in this section. The INPUT BULK DATA COUNT is the total number of lines in the current section of the input file, including blank lines and comments.

**Note**

The total number for the Main Bulk Data Section in this model is 25, while the total count for the SORTED BULK DATA ECHO is only 13. The first number is the total number of entries, including blank lines and comments, and the total count after the sorted listing is the number of actual input lines used.

Sorted bulk data for PARTs occurs after the main bulk data listing and is in the order in which it appears in the input file. For example, in the output from sample inrel2.dat (shown in “Inertia Relief Analysis Using Superelements”), the next output is for PART 1. Selected data from that output follows:
Similar output occurs for all PARTs defined in the input file.

### 8.2 Boundary Grid Search Output for PARTs

When using PARTs, the next section of the output is the output from the boundary grid search for each superelement. For this sample the output is:
When reading this output, you need to understand what the program is printing without being misled. This section of the output provides the list of BOUNDARY GRIDS for each PART as the grids are found during the boundary search. This search is performed one PART at a time, looking at all PARTs that have not yet been checked. The grid points listed are points that have not been identified before. For example, superelement 3 connects to residual structure points 19, 20, 41 and 42; however, because points 41 and 42 have already been identified as BOUNDARY GRIDS in superelement 2, they are not listed for superelement 3. This information can be found in the Individual Superelement Map, which is described later in this section.

### 8.3 Superelement Definition Table

The next output is a set of tables labeled SUPERELEMENT DEFINITION TABLE. Two of these tables are printed for each superelement run. Both of these tables provide a list of the superelements for the current run and a brief description of the type of superelement. Both of these tables provide the same information; however, the order in which the superelements appear in each table differs. In the first table, the superelements are sorted in numeric order; in the second table they are sorted by processing order.

The following tables come from the output generated by inrel2.dat.
### Superelement Tables

<table>
<thead>
<tr>
<th>SuperElement</th>
<th>SuperElement</th>
<th>Order</th>
<th>SuperElement</th>
<th>Type</th>
<th>Label</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>PRIMARY (BEGIN BULK SUPER)</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>0</td>
<td>2</td>
<td>0</td>
<td>PRIMARY (BEGIN BULK SUPER)</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>0</td>
<td>3</td>
<td>0</td>
<td>PRIMARY (BEGIN BULK SUPER)</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>0</td>
<td>4</td>
<td>0</td>
<td>PRIMARY (BEGIN BULK SUPER)</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>0</td>
<td>5</td>
<td>0</td>
<td>PRIMARY (BEGIN BULK SUPER)</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>0</td>
<td>6</td>
<td>0</td>
<td>PRIMARY (BEGIN BULK SUPER)</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>0</td>
<td>7</td>
<td>0</td>
<td>PRIMARY (BEGIN BULK SUPER)</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>8</td>
<td>0</td>
<td>RESIDUAL STRUCTURE</td>
<td></td>
</tr>
</tbody>
</table>

The first column in these tables contains the superelement number (SEID), and the second column contains the identification of the PRIMARY SUPERELEMENT for the current superelement.

The third column contains the PROCESS ORDER, or the numeric sequence for processing. In the first table, these numbers are most likely not in ascending order (this table is sorted on SEID); however, these numbers appear in ascending order in the second table.

In this model, the residual structure is defined in the Main Bulk Data Section and is labeled under TYPE as the residual structure. Any superelements that are primary superelements (those superelements that have geometry and elements, described in the chapter on secondary superelements) are labeled as either PRIMARY (defined in the Main Bulk Data Section) or PRIMARY (BEGIN BULK SUPER) (PARTs, defined in a BEGIN SUPER Section).

If repeated or mirrored components exist, you can use secondary superelements to model these components. If these components are used, the TYPE can appear as REPEATED (SEBULK) (PARTs) or as either IDENTICAL SE or MIRROR SE (image superelements).
Chapter 9: Introduction to Superelements in Dynamic Analysis

- Description of Dynamic Reduction Procedures
- Reduction Methods Used for Superelements
9.1 Description of Dynamic Reduction Procedures

This chapter describes the processes used in reducing superelements in dynamic analysis. There are a number of user options available with which you can control the solution process and accuracy. As shown in the preceding chapters, when superelements are used in a static solution, a static reduction process is performed to replace the physical superelement model with reduced matrices that have the same properties.

In dynamic analysis, a reduction must also be performed for all superelements; however, several options are now available to you. In static analysis, no approximation is introduced by using superelements. That is, the full static response of the superelement can be represented by the reduced matrices. In statics, the fixed-boundary solution, added on to the boundary solution ("Introduction and Fundamentals"), provides the exact solution to the problem.

Unfortunately, the reduction methods in dynamics are not exact, as they are in the static solutions. The reduction process in statics provides a solution that is written as

$$ U_o = U_o^0 + G_{oa} U_a $$

Equation 9-1.

In Eq. 9-1, the fixed-boundary solution, $U_o^0$ is known exactly. In dynamic reduction, the equivalent (exact interior motion) requires calculating all of the modes of each superelement, which is normally prohibitive. Therefore, approximate reduction techniques are used.

In dynamic reduction, there is the assumption that the solution for each superelement can be represented by using superposition of a series of shape functions.

$$ U_o = G_{ot} \cdot U_t + G_{oq} \cdot U_q $$

Equation 9-2.

where:

$U_o$ = The motion of the interior points of the current superelement.

$G_{ot}$ = The static transformation matrix (see "Introduction and Fundamentals").

$G_{oq}$ = The dynamic transformation matrix.

$U_t$ = the solution for the motion of the physical exterior DOFs.

$U_q$ = The solution for the motion of the generalized exterior DOFs (described in this chapter).

If this approach is compared to the one used in static analysis, the static part of the transformation ($G_{ot}$) is identical; however, instead of the fixed-boundary solution ($U_o^0$), we now have a dynamic
transformation matrix \( G_{oa} \) that must be multiplied by the motion of a set of generalized coordinates to obtain the solution for the interior DOFs of the superelement.

There are two classes of reduction used for superelements in dynamics: static reduction and dynamic reduction. Dynamic reduction is the more accurate approach for most problems, but it also requires more computational effort and some user judgement.

### 9.2 Reduction Methods Used for Superelements

The following information explains the internal operations used in NX Nastran when reducing superelement matrices for a dynamic solution. The user interface is shown in the next section.

#### Static Condensation (Guyan Reduction)

The default approach in superelement dynamic analysis is static condensation, which is better known as Guyan reduction since both stiffness and mass are reduced. In this approach, the dynamic transformation matrix \( G_{oa} \) is null, and only the static transformation matrix is used. The motion of the superelement is represented by Eq. 9-3:

\[
U_o = G_{ot} \cdot U_t
\]

**Equation 9-3.**

Or, for this case, \( G_{oa} \) is identical to \( G_{ot} \).

The stiffness, mass, damping, and loads applied to the superelement are transformed into the reduced matrices using only the static transformation matrix. For the mass and stiffness, this process is shown in Eq. 9-4 and Eq. 9-5:

\[
[K_{aa}] = [K_{aa}] + [K_{oa}]^T [G_{oa}]
\]

**Equation 9-4.**

\[
[M_{aa}] = [M_{aa}] + [M_{oa}]^T [G_{oa}] + [G_{oa}]^T [M_{oa}] + [G_{oa}]^T [M_{oo}] [G_{oa}]
\]

**Equation 9-5.**

It is assumed that the motion of the exterior DOFs, multiplied by the static transformation matrix, can represent the dynamic solution of the superelement and that local dynamic effects can be ignored. This assumption may be valid for a number of cases (very stiff components or cases where local dynamic effects are to be ignored), but often static reduction is not sufficient. Unfortunately, there is no built-in feature to tell you whether the reduction method you have used is correct for your problem. You must determine whether the approach you have selected is sufficient for the problem you are solving.

There are different options for defining the analysis set (a-set) for static condensation. The most direct method for an external superelement creation run is with the ASETi bulk entries. If BSETi and CSETi bulk entries are defined in this case, the software will assign the DOFs listed to the a-set when
static condensation is the superelement reduction method. Although, the fixed and free designations are meaningless when using static condensation.

**Note**

Unless dynamic reduction and all modes of the superelement are used, there is no guarantee that your solution is correct.

If a static reduction is used, you may be comfortable with the lowest modes of the system, but higher frequency modes or local modes can be either missed entirely or calculated incorrectly.

For most cases, it is safest to use a dynamic reduction for your superelements; however, you must determine what are reasonable cost/benefit ratios.

**Dynamic Reduction**

Dynamic reduction consists of using additional shape functions (\(G_{oq}\) in the DMAP) to improve the approximation used to reduce a superelement (over the approximation that exists if only a static reduction were used).

For reference, we will look at conventional dynamic analysis. For many problems, the mode shapes of a structure are used to transform the equations of motion into modal coordinates, thus simplifying the solution. If this approach and all modes of a structure are used, no approximation occurs in this transformation.

Dynamic reduction of superelements is similar to the transformation of a structure into modal coordinates. However, the transformation involves the use of both static and dynamic shapes.

The dynamic reduction method available when using superelements is Component Modal Synthesis.

As before, the static transformation matrix is generated. This operation is always performed.

In dynamics, a dynamic transformation can be performed in addition to the static transformation. A series of dynamic shape-functions (\(G_{oq}\) in DMAP) is found and used to dynamically transform the superelement properties to the exterior DOFs.

When doing a dynamic reduction, the program calculates dynamic shapes based on the boundary conditions you have selected.

**Note**

You choose the shapes used to perform the dynamic reduction. The accuracy of the final solution depends on how many shape functions you use and how these shapes are calculated.

At this point, we must discuss the subsets of the A-set. The A-set in NX Nastran is the set containing all exterior DOFs for a superelement. When a superelement is processed, the component matrices are first generated for the G-set, then the matrices go through a series of processing steps until what remains is a set of reduced matrices that attach to the A-set DOFs. If the reduction is performed properly, these reduced matrices contain sufficient information to represent the dynamic behavior of the superelement.

The A-set is divided into several subsets for dynamic reduction. The following subsets of the A-set are used:

- Physical exterior DOF of the superelement
Q Generalized coordinates in the A-set

Note

The A-set is divided into a number of subsets. These subsets determine whether an exterior DOF is held constrained during the calculation of the dynamic shape functions. Note that this has no affect on whether the points are constrained once the superelement has been processed.

For purposes of dynamic reduction, the T-set is divided into three subsets:

B Physical exterior DOF that are constrained during the calculation of the dynamic transformation vectors.

C Physical exterior DOFs that are not constrained during the calculation of the dynamic transformation vectors.

R Reference DOFs—these are treated the same as the C-set during dynamic reduction.

The B-, C-, and R-sets allow you control over the way the dynamic transformation vectors are calculated. These sets define how the exterior DOFs are treated during the calculation of the dynamic transformation matrix for the superelement. The placement of a DOF in a B-, C-, or R-set does not determine how the point is treated in downstream superelements or whether the point is constrained in the final solution.

By default, all physical exterior DOFs for a superelement are placed in the B-set for that superelement. If you want the DOFs to be treated differently, you must place the DOFs in either the C- or R-set (see “Input and Output for Dynamic Reduction”).

Note

Placing an exterior DOF in the B- or C- or R-set for a superelement determines how that DOF is treated during the calculation of the dynamic shape functions for the superelement. These sets do not apply physical constraints to the model and have no impact on how the DOF is treated in downstream superelements.

Because the motion of a superelement is represented as a linear combination of the vectors used in the reduction, these vectors should resemble (as closely as possible) the actual deformations the superelement experiences in the final dynamic solution.

To illustrate, we will look at a simple problem—the vibration modes of a cantilever beam—using several different approaches. The model is shown in Figure 9-1.
We divide the beam into two superelements using the following SESET entries (it will also be done using PART superelements):

SESET,1,7,THRU,11
SESET,2,2,THRU,5

Using this definition, we find the first four system modes, using one, two, and three component modes per superelement.

We look at this problem in several ways—first by placing all exterior DOF in the B-set (the default), then by placing all exterior DOFs in the C-set, and finally, by placing point 6 (which is exterior to superelement 2) in the C-set for superelement 2.

The input files are not shown here because we have not yet discussed the format; however, the results are shown.

The following table shows the first four natural frequencies (in-plane bending only) for this model. The first set of results represents a model with no superelements (the correct finite element solution); the second set of solutions represents the results found by using static reduction only; the third set represents those found by placing all exterior DOFs in the B-set (fixed boundary); the fourth set shows the results obtained by placing all exterior DOFs in the C-set (free-free); and the fifth set shows the results found by placing all exterior DOFs for superelement 1 in the B-set, while placing grid point 6 in the C-set for superelement 2. In the results for each set of boundary conditions, the first column represents the solution found by using only one mode for each superelement; the second column represents the results using two modes for each superelement, and so on.

<table>
<thead>
<tr>
<th>Baseline</th>
<th>Static Reduction</th>
<th>Fixed Boundary</th>
<th>Free Boundary</th>
<th>Mixed Boundary</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>one mode/se</td>
<td>two mode/se</td>
<td>three mode/se</td>
</tr>
<tr>
<td></td>
<td></td>
<td>one mode/se</td>
<td>two mode/se</td>
<td>three mode/se</td>
</tr>
<tr>
<td></td>
<td></td>
<td>one mode/se</td>
<td>two mode/se</td>
<td>three mode/se</td>
</tr>
<tr>
<td></td>
<td></td>
<td>one mode/se</td>
<td>two mode/se</td>
<td>three mode/se</td>
</tr>
</tbody>
</table>

Figure 9-1. Cantilever Beam Model for Dynamic Reduction
How the exterior DOFs are handled during the calculation of the component modes can make a significant difference in the accuracy of the results, as shown by the results in the above table.

Using only static reduction is the same as if an A-set were defined for the model (without superelements), including only grid point 6. Surprisingly, this approach gives a reasonable estimate of the first mode, and a somewhat reasonable estimate of the second mode (within 35% of the correct value).

Note

For some models static reduction may be all that is needed.

When all exterior points are in the B-set, the component modes of superelement 1 are found with grid point 6 constrained, or cantilevered, and the component modes of superelement 2 are found with grid points 1 and 6 fixed, or in a fixed-fixed state. The results for this approach are accurate for this model. With only one mode for each superelement, the first two elastic modes of the model are predicted within 1%, and the third mode is predicted within 12%. When additional component modes are added, the results quickly converge to the correct solution. With three modes per superelement, the fourth system mode is predicted with an accuracy of better than 1%.

When all exterior points are in the C-set, the component modes of each superelement are found with no constraints applied on the exterior points, or (for this model) in the free-free state. The first system mode is predicted well (within 2%) using only one mode per superelement. Notice that when using only one mode per superelement, the model finds only three system modes, which is explained later in this section. For this model, predicting the second system mode using free-free CMS is difficult. Using three modes per superelement, the results for the second mode are only within 11% of the correct value, indicating how much impact the method used to handle the exterior points during CMS can have.

When the fourth approach is used, the component modes of superelement 1 are found with grid point 6 constrained (cantilevered), and the component modes of superelement 2 are found with grid point 1 fixed and no constraints on grid point 6 (cantilevered). These results are the best for all the methods shown.

Conclusion

We recommend dynamic reduction for almost all models using superelements in a dynamic solution. How you treat the exterior points during the dynamic reduction calculation is very important to the accuracy of your solution. We recommend calculating component modes using the boundary condition that best approximates how you expect the component to behave when combined with the rest of the structure.

Fixed-Boundary Dynamic Reduction

The default method used in NX Nastran when performing a dynamic transformation of a superelement is the fixed-boundary approach, where all exterior DOFs belong to the superelement's B-set. If this
approach is used, all exterior DOFs are held fixed during the calculation of the dynamic transformation vectors. When component modal synthesis is used, the default method is better known as the Craig-Bampton approach, which is the most commonly used approach.\(^1\)

**Description of Methodology (Better Known as Craig-Bampton CMS)**

After applying any multipoint constraints and all physical constraints to the superelement, the component matrices have been reduced to the F-set. These (F-set) superelement matrices are partitioned into two sets of degrees-of-freedom (DOFs). The first set (the B-set) represents the boundary (exterior) points. The second set is the remaining interior DOFs (the O-set).

At this point, a set of constraint modes is generated. Each constraint mode represents the motion of the superelement resulting from moving one boundary DOF 1.0 unit while holding the other boundary DOFs fixed. Therefore, there is one constraint mode for each boundary DOF. (These vectors are known as \(G_{OT}\) in NX Nastran.)

In matrix form

\[
\begin{bmatrix}
K_{oo} & K_{ob} \\
- & - \\
K_{bo} & K_{bb}
\end{bmatrix}
\begin{bmatrix}
\phi_{ob} \\
- \\
I_{bb}
\end{bmatrix}
= 
\begin{bmatrix}
0 \\
- \\
P_b
\end{bmatrix}
\]

Equation 9-6.

\((P_b\) is not actually applied.\)

The first line of this expression gives

\[
\{\phi_{ob}\} = -[K_{oo}]^{-1}[K_{ob}]\{I_{bb}\}
\]

Equation 9-7.

providing the following constraint modes:

\[
\{\phi_b\} = \begin{bmatrix}
\phi_{ob} \\
- \\
I_{bb}
\end{bmatrix}
\]

---

The transformation matrix in NX Nastran is known as $G_{OA}$. This matrix has one column for each A-set DOF and one row for each O-set DOF. This matrix contains two submatrices, $G_{OT}$ and $G_{OQ}$. $G_{OT}$ is the static part of the transformation, and $G_{OQ}$ is the dynamic transformation. For example, $G_{OT}$ contains the static transformation vectors in columns associated with the T-set (physical exterior DOFs), while $G_{OQ}$ contains the dynamic transformation vectors in columns associated with the Q-set (generalized exterior coordinates).

Now the O-set equations are solved for the fixed-boundary modes $\{\Phi_{OO}\}$.

$$-\omega_k^2 [M_{oo}] \{\Phi_{oo}\} + [K_{oo}] \{\Phi_{oo}\} = 0$$

Equation 9-8.

(You can calculate as many fixed-boundary modes as you want.) Then the fixed-boundary modes are concatenated with the constraint modes to form the generalized coordinates.

$$\{\Phi_G\} = \begin{bmatrix} \Phi_{ob} & \Phi_{oo} \\ - & - \\ I_{bb} & 0 \end{bmatrix}$$

Equation 9-9.

The terms in the upper part of this matrix are stored in $G_{OT}$ and $G_{OQ}$. The lower part is not stored.

The mass and stiffness matrices are pre- and postmultiplied by these modes to obtain the generalized mass and stiffness.

$$[K_{aa}] = \{\Phi_G\}^T [K_{ff}] \{\Phi_G\}$$

$$[M_{aa}] = \{\Phi_G\}^T [M_{ff}] \{\Phi_G\}$$

Equation 9-10.

where the F-set is the union of the B- and O-sets.

These generalized matrices contain physical DOFs, representing the boundaries, and modal coordinates, representing the fixed-boundary component modes.

At this point, these matrices can be treated like any other structural matrices, and data recovery can be performed for the component in a manner similar to using modal coordinates. That is, the displacements of the generalized coordinates are multiplied by the associated transformation vectors and added together to obtain the physical displacements of the component.

The calculated component modes for each superelement are mass-normalized in NX Nastran (regardless of the scaling requested on the EIGR or EIGRL entry). If you ask the program to print the component modes, these modes are printed using the normalization requested on the EIGR or EIGRL entry.
Fixed-Boundary Component Modal Synthesis Sample

The following model is an example of fixed-boundary component modal synthesis. The model is a cantilever beam (considering axial deformation only) divided into two superelements.

![Diagram of Superelement 1 and 2](image)

Spring Stiffness = 1.

Each Mass = 1.

The following main bulk data entries are used to control the dynamic reduction. Though these entries are discussed in the next section, they are briefly discussed here to help you understand how NX Nastran does CMS. The SEQSET1 entries instruct the program to use SPOINTs 1001 and 1002 to represent the component modes for superelement 1, and SPOINT 1005 represents the component mode for superelement 2.

```
SESET,1,4,5
SESET,2,2
SPOINT,1001,THRU,1010
SEQSET1,1,0,1001,1002
SEQSET1,2,0,1005
```

**Theoretical Solution for Frequencies**

<table>
<thead>
<tr>
<th>i</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td>$f_i (\text{cps})$</td>
<td>0.0553</td>
<td>0.1592</td>
<td>0.2438</td>
<td>0.2991</td>
</tr>
<tr>
<td>$\lambda_i = \omega^2$</td>
<td>0.1206</td>
<td>1.0000</td>
<td>2.3473</td>
<td>3.5321</td>
</tr>
</tbody>
</table>

This problem uses a single-level tree, thus all exterior points from the superelements are interior points in the residual structure.

**Processing Superelement 1**

The physical model of superelement 1 is shown below, with interior points 4 and 5 and exterior point 3. The masses on points 4 and 5, along with the springs connecting these points, belong to this superelement. The mass at point 3 is not a part of this superelement, because point 3 is an exterior point, and concentrated masses are treated as elements when the bulk data is partitioned.

![Diagram of Superelement 1](image)

First, generate the G-sized stiffness and mass matrices. Because this superelement is a tip superelement, the KJJ and KGG are identical. (Similarly, MJJ and MGG are identical.) For this
superelement, the G-set consists of points 3, 4, and 5. (Although SPOINTs 1001 and 1002 are part of the G-set, these points are not shown yet.)

\[
K_{gg} = \begin{bmatrix} 1 & -1 & 0 \\ -1 & 2 & -1 \\ 0 & -1 & 1 \end{bmatrix} \quad M_{gg} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} u_3 \\ u_4 \\ u_5 \end{bmatrix}
\]

**Equation 9-11.**

Grid point 3 is the boundary point. Solve for the constraint modes.

\[
\begin{bmatrix} 1 & -1 & 0 \\ -1 & 2 & -1 \\ 0 & -1 & 1 \end{bmatrix} \begin{bmatrix} 1 \\ u_4 \\ u_5 \end{bmatrix} = \begin{bmatrix} P_b \\ 0 \\ 0 \end{bmatrix}
\]

**Equation 9-12.**

Where (using equations $10^{-6}$, $10^{-7}$):

\[
K_{oo} = \begin{bmatrix} 2 & -1 \\ -1 & 1 \end{bmatrix}
\]

\[
K_{ob} = \begin{bmatrix} -1 \\ 0 \end{bmatrix}
\]

\[
K^{-1}_{oo} = \begin{bmatrix} 1 & 1 \\ 1 & 2 \end{bmatrix}
\]

\[
\Phi_{ob} = -\begin{bmatrix} 1 \\ 1 \end{bmatrix} \begin{bmatrix} -1 \\ 0 \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \end{bmatrix}
\]

\[
\Phi_b = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} \begin{bmatrix} U_3 \\ U_4 \\ U_5 \end{bmatrix}
\]

This matrix states that if point 3 moves one unit, points 3, 4, and 5 will all move 1.0 units. This statement makes sense if you look at the physical model of superelement 1. No constraints are applied to superelement 1; therefore, when point 3 is moved statically, the superelement moves as a rigid body. Once we look at the dynamic effects, there are additional shape functions to handle the elastic response.

Now we solve for fixed-boundary modes.
\[
\begin{bmatrix}
-\omega^2 M_{oo} + K_{oo} \end{bmatrix} \{ \phi_{oo} \} = 0. = \begin{bmatrix}
-\omega^2 & 0 \\
0 & -\omega^2 
\end{bmatrix} \begin{bmatrix} 2 & -1 \\
-1 & 1 
\end{bmatrix} \{ \phi_{oo} \}
\]


\[
det \begin{bmatrix}
2 - \omega^2 & -1 \\
-1 & 1 - \omega^2 
\end{bmatrix} = 0 \quad \omega^2 = .3819, 2.618 \\
\mu^2 = .098 \, \text{Hz,} \quad .2575 \, \text{Hz}
\]

\[
\phi_1 = \begin{bmatrix} .6180 \\
1.0 
\end{bmatrix} \quad \text{Normalized to Maximum disp} \\
\phi_2 = \begin{bmatrix} -1.0 \\
.6180 
\end{bmatrix} \\
\phi_{oo} = \begin{bmatrix} .5257 \\
-.8506 \\
.8506 \\
.5257 
\end{bmatrix}
\]

The above matrix is intended to show that the modes are calculated and used internally scaled to unit generalized mass. If you ask the program to print the eigenvectors (DISP = ALL in case control), the eigenvectors are printed using the normalization requested on the EIGR or EIGRL entry.

At this point, we have the transformations for superelement 1. Combining the static and dynamic transformations, we obtain the set of generalized transformation vectors and will finish processing the superelement. The transformation matrix shown in Eq. 9-14 has the three transformation vectors in it. (Internally, the transformation matrix is stored as \( G_{ot} \) and \( G_{oq} \), and the rows corresponding to the exterior DOFs are not stored.)

\[
\phi_G = \begin{bmatrix}
1 & 0 & 0 \\
1 & .5257 & -.8506 \\
1 & .8506 & .5257 
\end{bmatrix} \begin{bmatrix} u_3 \\
u_4 \\
u_5 
\end{bmatrix}
\]

Equation 9-14.

The transformation matrix (as shown) has the static transformation plus two modal transformation vectors. Each modal transformation vector is associated with one Q-set DOF (hence the name SEQSET1). The second column in the above transformation states that if DOF 1002 is moved 1.0 units, point 3 does not move, point 4 moves .5257 units, and point 5 moves .8506 units.

The superelement matrices are now reduced to the exterior DOFs using the transformation matrix.
Once again, DOFs 1001 and 1002 are scalar points used to represent the modes of Superelement 1.

First notice that the physically reduced stiffness (the 1-1 term) is 0.0 for this superelement. This superelement is a special case, where the interface between the superelement and the rest of the structure is determinate. In this case the reduced stiffness is 0.0 (as shown in Eq. 9-15). Look at the static transformation vector. If point 3 moves 1.0 units, then points 4 and 5 also move 1.0 units. The static transformation is a rigid-body vector. The reaction at the interface when the structure moves in this shape is 0.0, indicating that the reduced stiffness is null. Statically, there is no reaction force when the interface moves, but this does not mean that the superelement is not connected to the rest of the structure.

Also notice that there are no coupling terms between the modal and physical DOFs in the stiffness matrix, showing that if the superelement moves (or is loaded) statically, the modes are not excited. The coupling between modal and physical DOFs occurs in the mass matrix (which is not used in statics), showing that if the interface is moved in a dynamic analysis, the modes respond dynamically.

**Processing Superelement 2**

The physical model of superelement 2 is shown below. This model has interior point 2, with its associated mass and the springs connected to it. This model also has exterior points 1 and 3. (Once again, the mass of these exterior points is not included in this superelement.)

Once again, the first step is to create the G-sized matrices, KGG and MGG. (KJJ and MJJ are identical to these matrices, because this is a tip superelement.)
\[ K_{gg} = \begin{bmatrix} 1 & -1 & 0 \\ -1 & 2 & -1 \\ 0 & -1 & 1 \end{bmatrix} \quad M_{gg} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix} \]

**Equation 9-16.**

These matrices are in terms of physical DOFs 1, 2, and 3.

**Note**

SPOINT 1005, used on the SEQSET1 for this superelement, is also a part of the G-set; however, SPOINT 1005 is not yet shown.

This superelement has two exterior points (1 and 3), thus the static transformation matrix contains the associated transformations. The static transformation matrix for this superelement is shown in Eq. 9-17.

\[
\phi_{b} = \begin{bmatrix} u_1 \\ u_2 \\ u_3 \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 1/2 & 1/2 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \\ u_3 \end{bmatrix} = \begin{bmatrix} u_1 \\ u_2 \\ u_3 \end{bmatrix}
\]

**Equation 9-17.**

The first column of this matrix shows the static motion of this superelement if DOF 1 is moved 1.0 units, while DOF 3 is constrained. Column 2 shows the motion if DOF 3 is moved one unit, while DOF 1 is constrained.

Now we solve the fixed-boundary eigenvalue problem. If we constrain DOFs 1 and 3, the remaining DOF is DOF 2, with a mass of 1.0 and a stiffness of 2.0. The results are shown in Eq. 9-18:

\[
\left\{ \phi_{oo} \right\} = \{1\} \quad u_2 = 2.0 \\
\quad \omega^2 = 2.0 \\
\quad f = .2251
\]

**Equation 9-18.**

Once again, the mode is normalized to unit modal mass. (This mode is identical if normalized to a maximum displacement of 1.0.) This mode is also used as the dynamic transformation vector. This modal DOF is associated with the superelement's Q-set (DOF 1005). The resulting transformation matrix is shown in Eq. 9-19.
\begin{equation}
\{ \phi_G \} = \begin{bmatrix}
1 & 0 & 0 \\
1/2 & 1 & 2 \\
0 & 1 & 0
\end{bmatrix} \begin{bmatrix}
u_1 \\
u_2 \\
u_3
\end{bmatrix}
\end{equation}

\textbf{Equation 9-19.}

As before, the labels above the columns represent the exterior DOFs of the superelement, and the terms in each column represent the motion of the superelement if that exterior DOF is moved 1.0 units.

Using this transformation, the reduced matrices for superelement 2 are as shown in \textbf{Eq. 9-20:}

\begin{equation}
\{ \phi_G \}^T [K_{gg}] \{ \phi_G \} = \begin{bmatrix}
.5 & -.5 & 0 \\
-.5 & .5 & 0 \\
0 & 0 & 2.0
\end{bmatrix} \begin{bmatrix}
u_1 \\
u_3 \\
u_{1005}
\end{bmatrix}
\end{equation}

\begin{equation}
\{ \phi_G \}^T [M_{gg}] \{ \phi_G \} = \begin{bmatrix}
.25 & .25 & .50 \\
.25 & .25 & .50 \\
.50 & .50 & 1.0
\end{bmatrix} \begin{bmatrix}
u_1 \\
u_3 \\
u_{1005}
\end{bmatrix}
\end{equation}

\textbf{Equation 9-20.}

Once again, DOF 1005 is a scalar point used to represent the mode of superelement 2.

\textbf{Processing the Residual Structure}

The physical model of the residual structure is shown below. Only points 1 and 3 (with their associated masses and constraints) and matrices representing superelements 1 and 2 remain. The residual structure also contains SPOINTs 1001, 1002, and 1005, which represent the superelement component modes; however, these SPOINTs have no physical location in space, making them difficult to visualize.

First, we generate the physical stiffness and mass matrices (KJJ and MJJ). Because the spring elements are in superelements 1 or 2, the physical stiffness matrix for superelement 0 is null. The masses on points 1 and 3 finally show up in the residual structure.
Note that the residual structure contains the physical DOFs associated with points 1 and 2 and the generalized DOFs representing the upstream superelement modes on SPOINTs 1001, 1002, and 1005.

Now that we have the physical matrices, we need to add the reduced matrices from the upstream superelements. First, we add the matrices from superelement 1, which is accomplished by adding the terms from the reduced matrices to the existing terms in the matrices for the associated DOFs. At this point we have intermediate matrices, which are shown in Eq. 9-22:

\[
K_{gg} = \begin{bmatrix}
0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0.3819 & 0 & 0 \\
0 & 0 & 2.618 & 0 & 0 \\
0 & 0 & 0 & 0 & 0
\end{bmatrix}
\]

\[
M_{gg} = \begin{bmatrix}
1 & 0 & 0 & 0 & 0 \\
0 & 3 & 1.3764 & -.3249 & 0 \\
0 & 1.3764 & 1.0 & 0 & 0 \\
0 & -.3249 & 0 & 1.0 & 0 \\
0 & 0 & 0 & 0 & 0
\end{bmatrix}
\]

Now, we add the reduced matrices from superelement 2. Once again, we add the terms from the reduced matrices to the existing terms in the locations associated with the exterior DOFs from superelement 2.

We now have the assembly matrices for the residual structure.
We are ready to process the residual structure. Start by applying the constraint at DOF 1.

Solve the residual structure eigenvalue problem

$$\left\{ K_{ff} - \omega^2 M_{ff} \right\} \{ \phi_f \} = 0$$

which gives $\omega^2 = 0.1206, 1.00, 2.3473, 3.5321$.

These values are the correct eigenvalues for this model. In this sample, all component modes are found for each superelement, so instead of a reduction (which is what is commonly done when using
superelements), this problem uses an exact transformation, and no approximation is introduced during superelement processing.

**Note**

If all modes of a superelement are used in the reduction, no approximation is introduced by using dynamic reduction. For most models, finding all modes of each superelement is prohibitive; therefore, an approximation is introduced by using a limited set of nodes.

This solution also gives the following eigenvectors for the residual structure, in terms of DOFs 3, 1001, 1002, and 1005.

\[
\begin{pmatrix}
\phi_1 & \phi_2 & \phi_3 & \phi_4 \\
.4285 & -.5773 & -.2280 & -.6565 \\
.2722 & -1.2858 & .3748 & -1.0132 \\
-.0067 & -.1159 & .6423 & .8243 \\
.0137 & -.2887 & .7705 & -.7568
\end{pmatrix}
\begin{pmatrix}
u_3 \\
u_{1001} \\
u_{1002} \\
u_{1005}
\end{pmatrix}
\]

Equation 9-26.

Now, we begin data recovery. For convenience, we will look only at the first eigenvector. For the physical DOFs in the residual structure, eigenvector 1 is

\[
\text{Residual Structure } \phi = \begin{pmatrix} 0 \\ .4285 \\ u_1 \\ u_3 \end{pmatrix}
\]

Equation 9-27.

The residual structure also contains the SPOINTs (1001, 1002, and 1005), but only the physical DOFs are shown here.

**Performing Data Recovery for Superelement 2**

The first step in performing data recovery for superelement 2 is getting the solution at the exterior points. The first eigenvector for the exterior DOFs of superelement 2 is shown in Eq. 9-28.

\[
\text{for exterior points } \phi_{2G} = \begin{pmatrix} 0 \\ .4285 \\ .0137 \\ u_1 \\ u_3 \\ u_{1005} \end{pmatrix}
\]

Equation 9-28.

The exterior DOFs include the modal DOF, represented by SPOINT 1005.
Now, we multiply the solution at the exterior DOFs by the transformation matrix to get the eigenvectors for superelement 2.

\[
\phi_{21} = \{\phi_{G2}\} \{\phi_{2G}\} = \begin{bmatrix} 1 & 0 & 0 \\ 1/2 & 1/2 & 1 \\ 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} 0 \\ .4285 \\ .0137 \end{bmatrix} = \begin{bmatrix} 0 \\ .2280 \\ .4285 \end{bmatrix}
\]

Equation 9-29.

Repeating the Same Process for Superelement 1

The solution at the exterior DOFs for eigenvector 1 is

\[
\text{for exterior points } \phi_{1G} = \begin{bmatrix} .4285 \\ .2722 \\ -.0067 \end{bmatrix}
\]

\[
\begin{bmatrix} u_3 \\ u_{1001} \\ u_{1002} \end{bmatrix}
\]

Equation 9-30.

Once again, multiply the solution at the exterior DOFs by the transformation matrix to get the eigenvector for superelement 1.

\[
\phi_{11} = \{\phi_{G1}\} \{\phi_{1G}\} = \begin{bmatrix} 1 & 0 & 0 \\ 1 & .5257 & -.8506 \\ 1 & .8506 & .5257 \end{bmatrix} \begin{bmatrix} .4285 \\ .2722 \\ -.0067 \end{bmatrix} = \begin{bmatrix} .4285 \\ .5773 \\ .6565 \end{bmatrix}
\]

\[
\begin{bmatrix} u_3 \\ u_4 \\ u_5 \end{bmatrix}
\]

The above data recovery provides the correct eigenvector for the entire structure. This eigenvector is normalized to unit generalized mass, which is the default in the eigenvalue solvers. If the eigenvectors found at the residual structure are normalized to unit mass, the eigenvectors for the entire system found during data recovery are normalized to unit mass, regardless of the reduction approach or scaling used during superelement reduction.

A sample in the next section shows the input and output from NX Nastran for this model.

Free-Free Dynamic Reduction

If all exterior points are placed in the C-set for a superelement, then the dynamic reduction is performed using a free-free approach. In this case, free-free states that none of the exterior points are held fixed during the calculation of the dynamic transformation vectors. If any constraints are applied on interior points, these constraints are included when performing the dynamic reduction.

The internal operations performed for free-free reduction are more complicated than those for fixed-boundary reduction. When a fixed-boundary reduction is performed, the dynamic transformation vectors are independent of the static transformation vectors, because the exterior points are held constrained while the dynamic transformation vectors are found. When a free-free reduction is
performed, one or more of the dynamic transformation vectors may be (actually, probably will be) a linear combination of the static vectors.

If there are no internal constraints, the rigid-body modes of a free-free component can be used as an example. These shapes are a linear combination of the static transformation vectors. More simply, the static transformation vectors are capable of describing any possible rigid-body motion of the component.

If any dynamic transformation vector that is a linear combination of the static vectors (or of any of the transformation vectors, for that matter) is used, the reduced matrices will be singular, and the problem will fail. Three methods of preventing this problem are provided in NX Nastran:

- Do not calculate the rigid-body modes. Simply do not request the rigid-body eigenvectors during the calculation of the dynamic transformations. (If employing CMS, use a value greater than 0.0 for the lowest frequency of interest.)

- Calculate the rigid-body modes, and hope that NX Nastran will remove them. Logic is included in the program that attempts to remove any dynamic transformation vectors that are a linear combination of the static transformations. This logic is shown later in this section.

- Calculate the rigid-body modes, but remove them manually using a SESUP or SUPORT (PARTs) entry (described in the next section). Any exterior DOFs defined on a SESUP entry are not constrained during the calculation of the dynamic transformation vectors. For each DOF listed on an SESUP, NX Nastran throws away one dynamic transformation vector, starting with the first one (lowest frequency). We do not recommend this approach, because the program does not check to verify that the thrown away vectors are actually a linear combination of the static transformation vectors. Therefore, elastic modes may be removed unintentionally.

Of the three approaches, not calculating the vectors is the safest.

The internal calculations and an example are shown after the description of mixed boundary dynamic reduction.

**Mixed-Boundary Dynamic Reduction**

The mixed-boundary dynamic reduction is used whenever exterior DOFs are placed in both the B- and C- and/or R-sets. The eigenvalue problem is now solved, with some (B-set) exterior DOFs constrained and others (C- and R-sets) not constrained. If this approach is used, the problem of dynamic transformation vectors that are a linear combination of the static transformations, (described in the preceding section on free-free dynamic reduction) still exists.

**CMS with Exterior DOF in the C- and/or R-SET**

When mixed-boundary or free dynamic reduction is performed, the F-set is partitioned into the V-set and the B-set. The B-set has the same definition as before—exterior DOFs that are constrained during the calculation of the dynamic transformation vectors.

The static transformation is calculated similarly for exterior DOFs, whether they are in the B-, C-, or R-set (subsets of the T-set); therefore, the matrix $G_{OT}$ is independent of the set into which the exterior DOFs are placed.
The calculation of the dynamic transformation depends on the set to which the exterior DOFs belong. If there are any exterior DOFs placed in either the C- or R-set, the program creates a V-set of DOFs for the dynamic reduction step.

The V-set is a combined set for use in dynamic reduction. The V-set consists of all DOFs in the F-set that do not belong in the B-set, therefore, it contains the O-, C-, and R-sets. Any DOF in the V-set is not constrained while calculating the dynamic transformation vectors. The stiffness and mass matrices are partitioned into the V- and B-sets. The V-set stiffness matrix has the form shown in Eq. 9-31.

\[
K_{vv} = \begin{bmatrix}
K_{oo} & K_{or} & K_{oc} \\
K_{ro} & K_{rr} & K_{rc} \\
K_{co} & K_{cr} & K_{cc}
\end{bmatrix}
\]

**Equation 9-31.**

The V-set mass matrix is similar.

The eigenvalue problem is set up for the V-set.

\[
[K_{vv} - \lambda M_{vv}][\phi_{vz}] = 0
\]

**Equation 9-32.**

This eigenvalue problem is used by a real eigenvalue method (EIGR or EIGRL) to find the dynamic transformation vectors. The program finds \(N_z\) vectors, where \(N_z\) is the number of vectors requested by the EIGR, or EIGRL Bulk Data entries. The matrix of dynamic transformation vectors, \(\phi_{vz}\), is the starting point for removing information from the dynamic transformation vectors that can be represented by the static transformation matrix.

The matrix \([\phi_{vz}]\) can be partitioned into the DOFs, represented in Eq. 9-33.

\[
[\phi_{vz}] = \begin{bmatrix}
\phi_{oz} \\
\phi_{rz} \\
\phi_{cz}
\end{bmatrix}
\]

**Equation 9-33.**

This matrix is partitioned into the O-set and A-set partitions. The A-set partition is augmented with 0.0 terms for the B-set to form \(\phi_{az}\).
\[ \begin{bmatrix} \phi_{oz} \\ \phi_{rz} \\ \phi_{cz} \\ \phi_{bz} \end{bmatrix} \]

**Equation 9-34.**

\([\varphi_{oz}]\) and \([\varphi_{az}]\) are stored for use in data recovery (printing the component modes if PARAM, FIXEDB, -1). In dynamic analysis, if PARAM, FIXEDB is set to -1, the program does not calculate the residual structure solution. Instead, it processes all superelements and performs data recovery on the component modes using \([\varphi_{oz}]\) and \([\varphi_{az}]\) which are scaled based on the request on the EIGRL or EIGR entry.

**Note**

Once a run using FIXEDB = -1 is complete, a restart with the value of FIXEDB changed to 0 (zero), which is the default, will complete the residual structure solution and data recovery using the assembly solution.

At this point, some of the dynamic transformation vectors may be a linear combination of the static transformation vectors. Because keeping these vectors would result in a singular matrix during further processing, it is necessary to remove any dynamic transformation vectors that are a combination of the static vectors.

This operation is accomplished inside subDMAP RESVEC. (Previously, a DMAP Module INREL did this.) The procedure begins using the eigenvectors provided by the READ module. Any rigid body modes are implicitly included in the static boundary transformation (\(G_{OT}\)). If you use the SESUP entry (not recommended, unless you are sure of the number of rigid-body modes in advance), the number of rigid body modes, \(N_r\), is determined by the number of R-set degrees-of-freedom, and the first \(N_r\) columns of \([\varphi_{vz}]\), which is called \(\varphi^0_{vz}\), are discarded.

\[ \begin{bmatrix} \phi_{vz} \end{bmatrix} = \begin{bmatrix} 1 \\ \frac{1}{\phi_{oz}} \\ \frac{1}{\phi_{rz}} \\ \frac{1}{\phi_{cz}} \end{bmatrix} \]

discarded

**Equation 9-35.**

Note that the program does not verify that these are rigid-body modes, and no messages are issued indicating that this operation has been performed.

The dynamic transformation vectors are then modified such that the motion of the C- and R-set variables is 0.0. This modification is accomplished by using the static transformation matrix, resulting in \(G^1_{oz}\) in Eq. 9-36 and Eq. 9-37.
\[
\[ G_{oI} \] = \[ G_{ob} G_{oc} G_{or} \]
\]

Equation 9-36.

\[
\left[ G_{o2}^1 \right] = \left[ \phi_{o2}^1 \right] - \left[ G_{oc} \right] - \left[ \phi_{rz}^1 \right]
\]

Equation 9-37.

where \( [G_{o2}^1] \) is the dynamic transformation matrix, with all effects represented by the static transformations removed. If any column of \( [G_{o2}^1] \) has all terms less than the parameter EPSRC, the column is discarded. Once again, no information messages will be issued that indicate which columns are removed.

The program contains an option to concatenate additional shapes into the \( G_{OQ} \) matrix. These shapes represent the fixed-boundary static solution of the superelement under unit accelerations in each of the six directions. These vectors are called inertia relief modes.

Note

Any column of the dynamic transformation matrix with all terms less than PARAM,EPSRC will be removed. If further processing difficulties occur that can be traced back to the DOFs representing the dynamic transformations (the Q-set), try increasing the value of EPSRC.

If inertia relief modes are requested (PARAM,INRLM < 0), a matrix, \( V_g \), containing the six rigid-body vectors, is created by the VECPLOT module. This matrix is partitioned into the A- and O-set components.

\[
\left[ V_g \right] = \begin{bmatrix} V_o \\ V_a \\ 0 \end{bmatrix}
\]

Equation 9-38.

The inertia relief mode shapes are calculated.

\[
\left[ H_o^i \right] = \left[ K_{oo}^{-1} \right] \left[ M_{oo} \right] \left[ V_o \right] + \left[ M_{oa} \right] \left[ V_a \right]
\]

Equation 9-39.

These inertia relief modes are concatenated onto the dynamic transformation vectors to form \( G_{oq} \).
Depending on the number of Q-set DOFs, the \([G_{oq}]\) matrix is truncated or padded with null columns to provide \(N_q\) columns.

The total transformation matrix is formed by merging the dynamic and static components, and this matrix is reassembled whenever needed.

\[
[G_{oa}] = [G_{ot} : G_{oq}]
\]

**Equation 9-41.**

### The Dynamically Reduced Matrices

Once the transformation matrices are created, the matrix reduction process is identical.

Remember that the off-diagonal (coupling) terms between the static and dynamic DOFs in the stiffness matrix are null, so the generalized stiffness coefficients are formed from the dynamic transformation.

\[
[K_{qq}] = [G_{oq}]^T[K_{oo}][G_{oq}]
\]

**Equation 9-42.**

When the total boundary stiffness matrix is needed, it is formed from its partitions \(K_{qq}\) and \(K_t\) (the statically reduced stiffness).

\[
[K_{aa}] = \begin{bmatrix} K_{qq} & 0 \\ 0 & K_t \end{bmatrix}
\]

**Equation 9-43.**

There is no coupling between the static and dynamic DOFs in the stiffness matrix. The static and dynamic degrees-of-freedom are coupled in the mass matrix. The matrix is formed from the equations

\[
[M_{qq}] = [G_{oq}]^T[M_{oo}][G_{oq}]
\]

\[
[M_{qt}] = [G_{oq}]^T[M_{ot} + M_{oo}G_{oa}^t]
\]

**Equation 9-44.**

If virtual mass effects are to be included, they are added.
The total mass matrix is formed by

\[
[M_{aa}] = \begin{bmatrix}
M_{qq} & M_{qt} \\
M_{sq} & M_{tt}
\end{bmatrix}
\]

\textbf{Equation 9-45.}

The damping matrices \([B_{gg}]\) and \([K^A_{gg}]\) are reduced to the A-set by the MATREDU module, which performs the following operations:

1. Eliminate multipoint constraints.

\[
[B_{gg}] \geq \begin{bmatrix}
\tilde{B}_{nn} & B_{mn} \\
B_{mn} & B_{mm}
\end{bmatrix}
\]

\[
[B_{nn}] = [G_{mn}]^T [B_{mn} G_{mn} + B_{mn}] + [B_{mn}]^T [G_{mn}] + [B_{nn}]
\]

2. Eliminate single-point constraints.

\[
[B_{nn}] \geq \begin{bmatrix}
B_{ff} & B_{fs} \\
B_{sf} & B_{ss}
\end{bmatrix}
\]


\[
[B_{ff}] \geq \begin{bmatrix}
\tilde{B}_{aa} & B_{ao} \\
B_{oa} & B_{oo}
\end{bmatrix}
\]

\textbf{Equation 9-46.}

4. Perform the reduction using the (combined static and dynamic) transformation matrix.

\[
[B_{aa}] = [G_{oa}]^T [B_{oo} G_{oa} + B_{oa}] + [\tilde{B}_{oa}]^T [G_{oa}] + [\tilde{B}_{aa}]
\]

\textbf{Sample Problem Using Free-Free Component Modal Synthesis}

Using a similar model to the one we used before (with the addition of density on the elements), we will show how this model is processed using free-free component modal synthesis. In this model, ROD elements with density are used in place of the spring elements in the previous example.
As before, grid points 1 and 3 are assigned to the residual structure. However, now we place them in the C-set for each superelement. Therefore, we are performing free-free CMS. The SECSET1 entry is employed to perform free-free CMS, using the following format:

```
SECSET1,1,123456,ALL
```

This entry works only for superelements defined in the Main Bulk Data Section. More information on this entry is provided in the next section.

**Process Superelement 1**

Grid point 3 is the exterior point.

As before, we first solve for the static (or constraint) modes.

\[
M_{gg} = \begin{bmatrix}
1 & 0 & 0 \\
0 & 3 & 0 \\
0 & 0 & 2 \\
\end{bmatrix}, \quad K_{gg} = \begin{bmatrix}
1 & -1 & 0 \\
-1 & 2 & -1 \\
0 & -1 & 1 \\
\end{bmatrix}
\]

As before, we first solve for the static (or constraint) modes.

\[
\begin{bmatrix}
1 & -1 & 0 \\
-1 & 2 & -1 \\
0 & -1 & 1 \\
\end{bmatrix}\begin{bmatrix}
u_4 \\
u_5 \\
\end{bmatrix} = \begin{bmatrix} P_b \\
0 \\
0 \\
\end{bmatrix}
\]

Point 3 is exterior, and points 4 and 5 are interior. Partitioning the stiffness into interior and exterior DOFs, we obtain
\[
K_{oo} = \begin{bmatrix}
2 & -1 \\
-1 & 1
\end{bmatrix}
\]
\[
K_{ob} = \begin{bmatrix}
-1 \\
0
\end{bmatrix}
\]
\[
K_{oo}^{-1} = \begin{bmatrix}
1 & 1 \\
1 & 2
\end{bmatrix}
\]

and solve for the static transformation vector.
\[
\therefore \{ \Phi_{ob} \} = - \begin{bmatrix}
1 & 1 \\
1 & 2
\end{bmatrix} \begin{bmatrix}
-1 \\
0
\end{bmatrix} = \begin{bmatrix}
1 \\
1
\end{bmatrix}
\]

As before, if point 3 moves one unit (statically), then points 4 and 5 also move one unit.
\[
\Phi_b = \begin{bmatrix}
1 \\
1 \\
1
\end{bmatrix}
\]

Now we need to solve for the elastic modes of the superelement. Because point 3 is in the C-set, we will solve the V-set eigenvalue problem for the free-free modes.
\[
[- \omega^2 M_{vv} + K_v] \{ \Phi_v \} = 0
\]

Once again, the V-set is a combination of the C- plus R- plus O-sets; therefore, the free-free eigenequations take the following form.
\[
\begin{bmatrix}
-\omega^2 & 0 & 0 \\
0 & -3\omega^2 & 0 \\
0 & 0 & -2\omega^2
\end{bmatrix} + \begin{bmatrix}
1 & -1 & 0 \\
-1 & 2 & -1 \\
0 & -1 & 1
\end{bmatrix} \{ \Phi_{vv} \} = 0
\]

To solve this problem we need:
\[
\text{let } \begin{bmatrix}
1 - \omega^2 & -1 & 0 \\
-1 & 2 - 3\omega^2 & -1 \\
0 & -1 & 1 - 2\omega^2
\end{bmatrix} = \mathcal{C}
\]

which provides the eigenvalues of this problem:
\[ \omega^2 = 0.0, \ 0.6667, 1.5 \]

The first eigenvalue (0.0) is a rigid-body mode, which is a linear combination of the static vectors. (In this case, the rigid-body mode is identical to the static transformation vector) that needs to be removed. These eigenvectors (normalized to unit generalized mass) are

\[
\Phi_1 = \begin{bmatrix} -0.4082 \\ -0.4082 \\ -0.4082 \end{bmatrix}; \quad \Phi_2 = \begin{bmatrix} 0.5477 \\ 0.1826 \\ -0.5477 \end{bmatrix}; \quad \Phi_3 = \begin{bmatrix} 0.7303 \\ -0.3651 \\ 0.1826 \end{bmatrix}
\]

At this point, we have:

\[
\phi_{xz}^{1} = \begin{bmatrix} -0.4082 & 0.5477 & 0.7303 \\ -0.4082 & 0.1826 & -0.3651 \\ -0.4082 & -0.5477 & 0.1826 \end{bmatrix}
\]

Now we need to remove any information from the dynamic vectors that can be represented by the static transformation.

\[
G_{oz}^{1} = \begin{bmatrix} -0.4082 & 0.5477 & 0.7303 \\ -0.4082 & -0.1826 & -0.3651 \\ -0.4082 & -0.5477 & 0.1826 \end{bmatrix} - \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} \begin{bmatrix} -0.4082 & 0.5477 & 0.7303 \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & -0.3651 & -1.0954 \\ 0 & -1.0954 & -0.5477 \end{bmatrix}
\]

The first mode is null and is removed. The matrix is partitioned.

Now an additional filter is performed to remove any remaining vectors that might not be independent. First, we transform the o-set mass using \( \phi_{xz} \):

\[
M_{zz2} = \phi_{xz}^T M_{oo} \phi_{xz} = \begin{bmatrix} 2.8 & 2.4 \\ 2.4 & 4.2 \end{bmatrix}
\]

A scaling matrix is found using the diagonals of \( M_{zz2} \):

\[
\text{scale} = \left[ \text{diagonal} \ M_{zz} \right]^{1/2} = \begin{bmatrix} 0.5976 & 0. \\ 0. & 0.4879 \end{bmatrix}
\]

What remains of the eigenvectors as scaled by this:

\[
[\phi_{xz}][\text{scale}] = \phi_x
\]

The generalized mass is scaled also.
\[ M_{zz} = [\text{scale}]^T [M_{zz}] [\text{scale}] = \begin{bmatrix} 1. & 0.6998 \\ 0.6998 & 1. \end{bmatrix} \]

Now the final filter is done. The reduced mass matrix is passed through the DECOMP module.

\[ M_{zz} = \begin{bmatrix} L \\ d \\ L^T \end{bmatrix} \]

The terms on the matrix diagonal are compared to the values on the factor diagonal.

\[ \text{RATIO}_i = \frac{M_{zzi}}{d_i} \text{ for this problem, } [\text{Ratio}] = \begin{bmatrix} 1.0 & 0 \\ 0 & 1.96 \end{bmatrix} \]

If Ratio exceeds a filter (PARAM,RESVRAT-default=1.E8), the associated vector is removed. For this example, none exceeded the allowable ratio, and both vectors are kept.

A final orthogonalization is performed on the remaining vectors. First, the stiffness is transformed:

\[ K_{zz} = \phi_x^T k_{oo} \phi_x = \begin{bmatrix} 0.2381 & 0 \\ 0 & 0.3571 \end{bmatrix} \]

The eigenvalue problem

\[ [K_{zz} - \lambda M_{zz}] \phi_{zz} = 0 \quad \phi_{zz} = \begin{bmatrix} 0.6693 & -1.2296 \\ 0.4099 & 1.3387 \end{bmatrix} \]

is solved for all roots. (These eigenvalues are not printed.) The final transformation vectors are found by:

\[ G_{oq} = [\phi_x][\phi_{zz}] = \begin{bmatrix} -0.3651 & -0.4472 \\ -0.5477 & 0.4472 \end{bmatrix} \]

The transformation matrix is then:

\[ G_{oa} = \begin{bmatrix} G_{oi} \\ G_{oq} \end{bmatrix} = \begin{bmatrix} 1 & -0.3641 & -0.4472 \\ 1 & -0.5477 & 0.4472 \end{bmatrix} \cdot \Phi G_1 = \begin{bmatrix} u_3 & u_{1001} & u_{1002} \\ 1.0 & 0 & 0 \\ 1. & -0.3651 & -0.4472 \\ 1. & -0.5477 & 0.4472 \end{bmatrix} \]

Using this to transform the stiffness and mass provides:
\[
M_{AA} = \Phi_{G_1}^T M_{FF} \Phi_{G_1} = \begin{bmatrix}
6.0 & -2.1909 & -.4472 \\
-2.1909 & 1.0 & 0 \\
-.4472 & 0 & 1.0
\end{bmatrix}
\]

\[
K_{AA} = \Phi_{G_1}^T M_{FF} \Phi_{G_1} = \begin{bmatrix}
0 & 0 & 0 \\
0 & .1667 & 0 \\
0 & 0 & 1.0
\end{bmatrix}
\]

Process Superelement 2

As before, we generate the superelement matrices and the static transformation vectors.

\[
K_{gg} = \begin{bmatrix}
1 & -1 & 0 \\
-1 & 2 & -1 \\
0 & -1 & 1
\end{bmatrix};
M_{gg} = \begin{bmatrix}
1 & 0 & 0 \\
0 & 3 & 0 \\
0 & 0 & 1
\end{bmatrix};
\Phi_b = \begin{bmatrix}
1 \\
0 \\
1/2 \\
1/2 \\
0 \\
1
\end{bmatrix}
\]

We now solve for the free-free modes:

\[
[-\omega^2 M_{vv} + K_{vv}]\{\Phi_v\} = 0
\]

\[
\left\{ \begin{array}{c}
-\omega^2 & 0 & 0 \\
0 & -3\omega^2 & 0 \\
0 & 0 & -\omega^2
\end{array} \right\} + \left\{ \begin{array}{c}
1 & -1 & 0 \\
-1 & 2 & -1 \\
0 & -1 & 1
\end{array} \right\}\{\Phi_w\} = 0
\]

\[
\omega^2 = 0, 1, 1.6667
\]

The eigenvectors (normalized to unit mass) are

\[
\Phi_1 = \begin{bmatrix}
-.4472 \\
-.4472 \\
-.447
\end{bmatrix};
\Phi_2 = \begin{bmatrix}
.7071 \\
0 \\
-.7071
\end{bmatrix};
\Phi_3 = \begin{bmatrix}
.5477 \\
.3651 \\
.5477
\end{bmatrix}
\]

As before, we remove any motion that can be represented by the static transformation matrix.
\[
G_{OZ_2} = \begin{bmatrix}
.4472 & .7071 & .5477 \\
-.4472 & .0 & -.3651 \\
-.4472 & -.7071 & .5477
\end{bmatrix} - \begin{bmatrix}
1 & 0 \\
.5 & .5 \\
0 & 1
\end{bmatrix} \begin{bmatrix}
.4472 & .7071 & .5477 \\
-.4472 & -.7071 & .5477
\end{bmatrix} = \begin{bmatrix}
0 & 0 & 0 \\
0 & 0 & -.9126 \\
0 & 0 & 0
\end{bmatrix}
\]

The first two eigenvectors are a linear combination of the static transformation and are removed. Only a single eigenvector remains. It is normalized to unit generalized mass, and we have:

\[G_{oq} = \{-.5773\} \quad u_2\]

or transformation matrix

\[
G_{OA} = \begin{bmatrix}
1 & 1 & -.5773 \\
2 & 2 & \end{bmatrix}
\]

We now have our transformation matrix for superelement 2:

\[
\Phi_{G_2} = \begin{bmatrix}
u_1 & u_3 & u_{1006}
\end{bmatrix}
\begin{bmatrix}
1 & 0 & 0 \\
1 & 1 & -.5773 \\
0 & 1 & 0
\end{bmatrix}
\begin{bmatrix}
u_1 \\
u_2 \\
u_3
\end{bmatrix}
\]

Notice that the filtering removed both the rigid-body mode and one of the elastic modes. The remaining mode has motion only for grid point 2 and looks similar to the mode from the fixed-boundary CMS. This is not always so, but, once again, we are looking at a special case where we obtain all eigenvectors of the superelement. (Because there is only one independent eigenvector available, both methods found this eigenvector.)

Now we use the transformation to reduce the stiffness and mass matrix. The dynamic transformation vector is represented in the reduced model by SPOINT 1006.

\[
\left[\Phi_{G_2}\right]^T[K_{gg}][\Phi_{G_2}] = \begin{bmatrix}
1 & 1 & 0 \\
2 & 2 & 0 \\
2 & 2 & .6667
\end{bmatrix}
\]

\[
\left[\Phi_{G_2}\right]^T[M_{gg}][\Phi_{G_2}] = \begin{bmatrix}
1.75 & .75 & -.866 \\
.75 & 1.75 & -.866 \\
-.866 & -.866 & 1.0
\end{bmatrix}
\begin{bmatrix}
u_1 \\
u_3 \\
u_{1006}
\end{bmatrix}
\]
Residual Structure

Once again, the model of the residual structure is:

![Diagram of residual structure]

Equation 9-47.

All the remaining grid points are in this model (points 1 and 3). All elements have been placed in upstream superelements, and we have the two sets of reduced matrices, representing superelements 1 and 2. We also have scalar points 1001, 1002, and 1006, representing the modes from the superelements.

First, we form the physical mass and stiffness for the residual structure.

\[
K_{jj} = \begin{bmatrix}
0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 \\
\end{bmatrix}
\]

\[
M_{jj} = \begin{bmatrix}
1 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 \\
0 & 0 & 1 & 0 \\
0 & 0 & 0 & 0 \\
\end{bmatrix}
\begin{bmatrix}
u_1 \\
u_3 \\
u_{1001} \\
u_{1002} \\
\end{bmatrix}
\]

Add the reduced matrices from superelement 1:

\[
K_{gg} = \begin{bmatrix}
0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 \\
0 & .1667 & 0 & 0 \\
0 & 0 & 1.0 & 0 \\
0 & 0 & 0 & 0 \\
\end{bmatrix}
\begin{bmatrix}
u_1 \\
u_3 \\
u_{1001} \\
u_{1002} \\
u_{1006} \\
\end{bmatrix}
\]
Add the reduced matrices from superelement 2:

\[
M_{gg} = \begin{bmatrix}
1 & 0 & 0 & 0 & 0 \\
0 & 7.0 & -2.1909 & -0.4472 & 0 \\
0 & -2.1909 & 1.0 & 0 & 0 \\
0 & -0.4472 & 0 & 1.0 & 0 \\
0 & 0 & 0 & 0 & 0 \\
\end{bmatrix}
\begin{bmatrix}
u_1 \\
u_3 \\
u_{1001} \\
u_{1002} \\
u_{1006} \\
\end{bmatrix}
\]

\[
K_{gg} = \begin{bmatrix}
0.5 & -0.5 & 0 & 0 & 0 \\
-0.5 & 0.5 & 0 & 0 & 0 \\
0 & 0 & 0.6667 & 0 & 0 \\
0 & 0 & 0 & 1.0 & 0 \\
0 & 0 & 0 & 0 & 0.6667 \\
\end{bmatrix}
\begin{bmatrix}
u_1 \\
u_3 \\
u_{1001} \\
u_{1002} \\
u_{1006} \\
\end{bmatrix}
\]

Apply constraint at DOF 1:

\[
M_{gg} = \begin{bmatrix}
2.75 & 0.75 & 0 & 0 & -0.866 \\
0.75 & 8.75 & -2.1909 & -0.4472 & -0.866 \\
0 & -2.1909 & 1.0 & 0 & 0 \\
0 & -0.4472 & 0 & 1.0 & 0 \\
-0.866 & -0.866 & 0 & 0 & 1.0 \\
\end{bmatrix}
\begin{bmatrix}
u_1 \\
u_3 \\
u_{1001} \\
u_{1002} \\
u_{1006} \\
\end{bmatrix}
\]

Solve the eigenvalue problem:

\[
[K_{ff} - \omega^2 M_{ff}] \{ \Phi_f \} = 0 \quad \text{yields} \quad \omega^2 = 0.0468, 0.38, 0.856, 1.217
\]

Once again, because no reduction was done (all modes were calculated for each superelement), these results are correct. (There is no approximation introduced by using superelements when all modes of each superelement are used.)

The resulting residual structure eigenvectors are:
Physical eigenvectors for the residual structure are:

\[
\begin{bmatrix}
\Phi_{RESIDUAL} =
\begin{bmatrix}
0 & 0 & 0 & 0 \\
-0.27355 & -0.3085 & 0.2121 & -0.344 \\
\end{bmatrix}
\end{bmatrix}
\begin{bmatrix}
u_1 \\
u_3
\end{bmatrix}
\]

**Data Recovery for Superelement 1**

First, we obtain the solution at the exterior points, and then we multiply the solution by the transformation matrix to obtain the solution:

\[
[\Phi]_{S.E.1} = [\Phi G_1][\Phi 1G]
\]

\[
[\Phi 1G] =
\begin{bmatrix}
-0.27355 & -0.3085 & 0.2121 & -0.344 \\
0.2338 & -1.2041 & 0.5772 & -0.8731 \\
0.006 & 0.0845 & -0.5634 & -0.8614
\end{bmatrix} \begin{bmatrix}
u_3 \\
u_{1001} \\
u_{1002}
\end{bmatrix}
\]

\[
[\Phi]_{S.E.1} =
\begin{bmatrix}
1. & 0 & 0 & 0 \\
1. & -0.3651 & -0.4472 & 0.2338 & -1.2041 & 0.5772 & -0.8731 \\
1. & -0.5477 & 0.4472 & 0.006 & 0.0845 & -0.5634 & -0.8614
\end{bmatrix}
\begin{bmatrix}
-0.27355 & -0.3085 & 0.2121 & -0.344 \\
0.2338 & -1.2041 & 0.5772 & -0.8731 \\
0.006 & 0.0845 & -0.5634 & -0.8614
\end{bmatrix} \begin{bmatrix}
u_3 \\
u_{1001} \\
u_{1002}
\end{bmatrix}
\]

which gives us the physical eigenvectors for superelement 1.

\[
[\Phi]_{S.E.1} =
\begin{bmatrix}
-0.27355 & -0.3085 & 0.2121 & -0.344 \\
-0.3616 & 0.09343 & 0.2533 & 0.36 \\
-0.3989 & 0.3889 & -0.356 & -0.251
\end{bmatrix} \begin{bmatrix}
u_3 \\
u_4 \\
u_5
\end{bmatrix}
\]

**Data Recovery for S.E. #2:**

Repeating the process,

\[
[\Phi]_{S.E.2} = [\Phi G_2][\Phi 2G]
\]

\[
[\Phi 2G] =
\begin{bmatrix}
0 & 0 & 0 & 0 \\
-0.27355 & -0.3085 & 0.2121 & -0.344 \\
0.0179 & 0.3541 & 0.8311 & -0.6585
\end{bmatrix} \begin{bmatrix}
u_1 \\
u_3 \\
u_{1006}
\end{bmatrix}
\]
In contrast to Superelements in Dynamic Analysis

\[
[\Phi]_{\text{S.E.2}} = \begin{bmatrix}
1 & 0 & 0 \\
.5 & .5 & -.5773 \\
0 & 1 & 0
\end{bmatrix}
\begin{bmatrix}
0 & 0 & 0 & 0 \\
.27355 & -.3085 & .2121 & -.344 \\
.0179 & .3541 & .8311 & -.6586
\end{bmatrix}
\]

\[
[\Phi]_{\text{S.E.2}} = \begin{bmatrix}
0 & 0 & 0 & 0 \\
-.1471 & -.3587 & -.37385 & .2083 \\
-.27355 & -.3085 & .2121 & -.344 \\
-.3616 & .09343 & .2533 & -.36 \\
-.3989 & .3889 & -.356 & -.251
\end{bmatrix}
\begin{bmatrix}
u_1 \\
u_2 \\
u_3 \\
u_4 \\
u_5
\end{bmatrix}
\]

we obtain:

\[
[\Phi]_{\text{FULL BY HAND}} = \begin{bmatrix}
0 & 0 & 0 & 0 \\
-.1471 & -.3587 & -.37385 & .2083 \\
-.27355 & -.3085 & .2121 & -.344 \\
-.3616 & .09343 & .2533 & -.36 \\
-.3989 & .3889 & -.356 & -.251
\end{bmatrix}
\begin{bmatrix}
u_1 \\
u_2 \\
u_3 \\
u_4 \\
u_5
\end{bmatrix}
\]

which compares to the results from an NX Nastran run for this model.

\[
[\Phi]_{\text{NX Nastran}} = \begin{bmatrix}
0 & 0 & 0 & 0 \\
.1471 & -.3587 & .3737 & .2082 \\
.2735 & -.3085 & -.212 & -.344 \\
.3616 & .0933 & -.2533 & .3601 \\
.3989 & .3888 & .3559 & -.251
\end{bmatrix}
\begin{bmatrix}
u_1 \\
u_2 \\
u_3 \\
u_4 \\
u_5
\end{bmatrix}
\]

The basic idea of how superelements are processed in a dynamic solution has been presented. Once again, you have the option to either perform a static reduction (the default) or a dynamic reduction. If you choose a dynamic reduction, you can treat the exterior DOFs any way you want (constrained or not constrained) during the calculation of the dynamic transformation.

If you perform a dynamic transformation, the accuracy of the results depends on your skill and engineering insight. As you have seen, obtaining all modes for each superelement gives the exact answer. If you obtain fewer modes, you are introducing approximations into the problem. Both how you treat the exterior DOFs and the number of eigenvectors obtained have an effect on the solution quality. If you do not treat the exterior DOFs carefully, you can compensate with a large number of component modes. Likewise, if you do not calculate many component modes, proper treatment of the exterior DOFs can improve the accuracy.

Unfortunately, there is no set method that works best for all problems. The default method of dynamic reduction (Craig-Bampton CMS) works well for most problems; however, there will always be problems for which dynamic reduction does not work well.
The number of methods for calculating dynamic transformation vectors is unlimited. Many of these methods are used in industry. The best summary of the standard methods of calculation are in Craig’s text.²

Chapter 10: Input and Output for Dynamic Reduction

- Case Control for Dynamic Reduction
- Single-Level Dynamic Reduction
- Multilevel Dynamic Reduction
This chapter demonstrates how to control the dynamic processing of superelements, along with the output associated with dynamic reduction.

### 10.1 Case Control for Dynamic Reduction

In the previous chapter, a description of the process involved in dynamic reduction was provided. Now we will define the input that controls this process. The first part of this section deals with the case control, and the second part deals with the bulk data. For clarity, the Bulk Data Section is broken into two subsections—one for superelements defined in the *Main Bulk Data Section* and the other dealing with PARTs. The case control commands are the same whether you are using PARTs or main bulk data superelements.

Dynamic reduction is invoked when both the necessary case control commands and bulk data entries are present. If either set of information is missing, a static reduction will be performed.

#### Case Control for Dynamic Reduction

This part of the Guide discusses case control for dynamic reduction and requires knowledge of how case control works for superelements. If you have not read “Loads, Constraints, and Case Control in Static Analysis”, we recommend that you read that section first.

The case control for dynamic solutions is simply an extension of the case control for static solutions. In fact, if you want to perform static reduction (the default) on all superelements, you do not have to consider any new case control for dynamics.

If, however, you want to perform a dynamic reduction, then you must use the expanded case control in nearly all cases, with only one exception—when a problem uses the same eigenvalue (or dynamic reduction) solution control, all superelements and for the residual structure. This exception is a rare occurrence that happens primarily in small sample problems.

The dynamic reduction approach for each superelement is determined by commands in the first SUBCASE for that superelement. The METHOD command controls the dynamic process.

**Note**

The commands that control the dynamic reduction for a superelement must appear in the first SUBCASE referencing that superelement.

If you want to perform component modal synthesis on a superelement, you must have a METHOD command pointing to either an EIGR or EIGRL Bulk Data entry.

If you are solving for the assembly modes (only needed for SOL 103 or modal solutions), then there must be a METHOD command in each residual structure SUBCASE in the case control. Since the introduction of multiple boundary conditions, multiple residual structure eigenvalue problems can be solved in SOL 103. If you take advantage of this feature, you must have a different value for BC in each residual structure SUBCASE (to provide differentiation for the separate solutions on the database).

**Figure 10-1** contains an example Case Control Section that demonstrates the different dynamic reduction methods using the sample model.
If the input file has the correct bulk data entries, the above Case Control Section requests the following:

**SUBCASE 1** (Applies to superelements 1 and 2.) This SUBCASE requests component modal synthesis using eigenvalue METHOD 1.

**SUBCASE 2** (Applies to superelements 3 and 4.) Because there are no commands requesting a dynamic reduction, these superelements have a static reduction only.

**SUBCASE 3** (Applies to superelements 5 and 6.) This SUBCASE instructs the program to perform Component Mode Reduction for these superelements.

---

**Figure 10-1. Sample Case Control for Dynamic Reduction**
SUBCASE 4  
(Appplies to superelement 7). This SUBCASE is the most commonly used form, which performs component modal synthesis. In this case, the program has been instructed to use the eigenvalue solution described on EIGRL (or EIGR) 1 to calculate the modes of superelement 7.

SUBCASE 5  
(Appplies to the residual structure.) This SUBCASE instructs the program to use eigenvalue METHOD 2 to solve for the assembly modes at the residual structure.

10.2 Single-Level Dynamic Reduction

The bulk data entries for single- and multilevel dynamic reduction are similar. These entries are shown in different sections to emphasize the differences in the approaches. As mentioned before, the case control is independent of the type of tree used (single- or multilevel).

In a single-level dynamic solution, all the superelements attach directly to the residual structure. Therefore, the component modes are passed to the residual structure for the system solution. In multilevel dynamics, it is possible to create assemblies, where component modes from upstream superelements are coupled into a superelement to obtain the assembly modes, before the reduced matrices are passed to the residual structure.

In a single-level tree, the DOFs that represent the modes of the components must be interior to the residual and exterior to the superelements, which is accomplished using the following methods.

Bulk Data Entries for Single-Level Dynamic Reduction of Main Bulk Data Superelements

To perform a dynamic reduction on a superelement—in addition to the required case control commands—you need to define the DOFs that will represent the dynamic transformation vectors (component modes). You also need to define how you want to treat the exterior DOFs during the eigenvalue solution. (The default is fixed-boundary.)

The following bulk data entries are used when performing dynamic reduction of superelements defined in the Main Bulk Data Section.

SEQSET and/or SEQSET1  One or both of these entries can be used to define the DOFs that will represent the superelement dynamic transformation vectors (component modes) in subsequent processing. These entries point to SPOINTs and/or grid points that are interior to the residual. If the model is defined using only main bulk data superelements, grid points can be used. If any PARTs exist, then only SPOINTs can be used.

For each DOF listed on SESUP entries, one dynamic transformation vector (starting with the lowest natural frequency) is removed. Because the program performs no checking, we do not recommend this approach.
### SEBSET and/or SEBSET1
List of exterior DOFs that are placed in the superelement’s B-set. These DOFs are constrained during the calculation of the dynamic transformation. (Default = all exterior DOFs)

### SECSET and/or SECSET1
List of exterior DOFs that are placed in the superelement’s C-set. These DOFs are not constrained during the calculation of the dynamic transformation vectors. (Default = none)

### SESUP
List of exterior DOFs that are placed into the superelement R-set. These DOFs are not constrained during the calculation of the dynamic transformation vectors. However, for each DOF listed on the SESUP entry, one dynamic transformation vector (starting with the first one) is removed without any checking performed.

### SENQSET
List of the number of automatically generated Q-set DOFs for selected superelements or all superelements. This entry can be used only if there are one or more PARTs in the model. If used, this entry automatically generates internal DOFs for the Q-set for selected superelements. These DOFs are passed to the residual structure, with no coupling to downstream superelements until the residual structure is processed. Therefore, this entry may not be the best choice for multilevel dynamic analysis.

### GRID and/or SPOINT
You must define the degrees-of-freedom NX Nastran will use for the component modes (Note: If you use the SENQSET, this step is not required.)

For main bulk data superelements, component modes must be attached to either grid points or SPOINTs (with the exception that if PARTs exist, then grid points cannot be used; the SENQSET entry can be used for both main bulk data superelements and PARTs).

To attach the component modes to grid or SPOINTs, use the SEQSET and/or SEQSET1 entries. When using these entries, you identify the superelement with which you are working and the selected DOFs for the grids or SPOINTs (0) that represent the component modes. Make sure you select a sufficient number of DOFs to represent all of the component modes you are calculating, or the unrepresented modes will not be included in the solution. In a single-level process, the GRID and SPOINTs that represent the component modes must be interior to the residual structure.

If you use GRID points to represent the component modes for a superelement, these grids should have no attached structural elements. If they have structural elements attached, then you will encounter User Fatal Message 5290, indicating that you are attaching a structural element to a Q-set DOF.

```
*** USER FATAL MESSAGE 5290 (GPSP)
EXTERNAL ID = 1001 DOF = 1 IS IN THE Q-SET,
BUT HAS A STIFFNESS TERM BEFORE REDUCTION.
^^ USER FATAL MESSAGE 9032 (ERRPH1) - RUN IS TERMINATED
DUE TO MESSAGE(S) ABOVE. TO CONTINUE PROCESSING ALL SUPERELEMENTS,
INSERT PARAM,ERROR,0 INTO BULK DATA.
```

The above message indicates that grid point 1001, which is in the Q-set for the current superelement, has a stiffness term before the component modes are calculated and the run is terminated. Notice
that the message mentions that PARAM,ERROR can be used to continue the run. A value of 0 for
PARAM,ERROR instructs the program to terminate processing of the current superelement when a
FATAL message in encountered. However, the run will continue to process other superelements until
it can go no further. If you use this parameter, you should always look at the output file carefully to
verify that you did not encounter a FATAL error.

If your model is defined using main bulk data superelements only, then you can easily use SPOINTs
for the component modes. When a model is defined using main bulk data superelements only,
all SPOINTs are interior to the residual structure by definition and cannot be interior to any other
superelement, which is different from how a model with PARTs will behave. In a model with PARTs,
SPOINTs can be interior to any superelement.

Note

If a model is defined using main bulk data superelements only, then SPOINTS are (by
definition) interior to the residual structure.

The following lines show how SPOINTs can be used to define the Q-set DOFs for a superelement.

```
SPOINT,1001,thru,1010
SEQSET1,1,0,1001,thru,1010
```

The above code creates 10 SPOINTs (1001 through 1010) and instructs the program to use these
SPOINTs to represent the first 10 modes from superelement 10.

**Samples of Single-Level Dynamic Reduction for Main Bulk Data Superelements**

First, we will use a model of a cantilever beam to show the input for the different approaches.
This model is also used to generate the table comparing results of different reduction methods in
“Introduction to Superelements in Dynamic Analysis”. In that section, several runs were made for
each dynamic reduction approach, using a different number of modes calculated at the superelement
level. All of these runs are performed using SOL 103 (SEMODES).

The approaches we will use are:

- Static reduction only (seg10_a.dat)
- Fixed-boundary CMS (seg10_b1.dat)
- Free-free CMS (seg10_c1.dat)
- Mixed-boundary CMS (seg10_d1.dat)

The following listing is the model used for this example; this model is a cantilever beam, as shown
in Figure 9-1. The model is divided into two superelements (1 and 2) using SESET entries.
Superelement 1 has interior points 7 through 11, and superelement 2 has interior points 1 through
5. Both superelements have grid point 6 as their exterior point.

**File cantbeam.dat - Input Model for the Examples**

```
escet,1,7,thru,11
seset,2,1,thru,5
$PARAM,GRDPNT,0
SPC 1 1 123456 0.
PBAR 1 1 0.1 0.01 0.01 0.001 0.
```
In Section above component is defined to represent the component modes, only the static reduction is performed on each superelement. The program will issue UWM 9013 (as follows) for the superelements.

```
^^^ USER WARNING MESSAGE 9013 (SEMRM) - NO SEQSETI BULK DATA ENTRIES
HAVE BEEN SPECIFIED FOR THIS SUPERELEMENT EVEN THOUGH A METHOD COMMAND APPEARS
IN ITS SUBCASE.

THE METHOD COMMAND IS IGNORED AND ONLY A STATIC CONDENSATION
OF THE MASS IS CONSIDERED.
```

This message indicates that there is a METHOD command requesting CMS for the superelement. However, no DOFs are defined to represent the modes found, thus CMS will not be performed for this superelement.

The output for each superelement is the same as in a static solution. The residual structure output includes the eigenvalue solution output as follows:

In the above input, the case control METHOD command applies to all superelements (because there is no SUPER command, and the default is SUPER=ALL), which makes you think this run is a component modal synthesis (CMS) run. However, because no SEQSETI entries exist in the Bulk Data Section to define DOFs to represent the component modes, only the static reduction is performed on each superelement.
File seg10_b1.dat - Fixed Boundary CMS of the Superelements

The following file is used to perform CMS using fixed boundaries (exterior DOFs). The case control has been expanded to include a separate SUBCASE for each superelement (recommended), and each SUBCASE requests a different METHOD request.

The Bulk Data Section contains SEQSET1 and SPOINT entries, to provide DOFs to represent the component modes. The combination of the METHOD command in case control and the EIGRL (or EIGR), SEQSET1, and SPOINT entries in the bulk data tells the program how to calculate the component modes. This combination also tells the program which DOFs will be used to represent the component modes found.

When the program calculates the component modes, the exterior DOFs of each superelement are constrained by default, meaning that the modes of superelement 1 are calculated with grid point 6 constrained, and the modes of superelement 2 are calculated with grid points 1 and 6 constrained. (Grid point 1 is an interior point with an SPC applied, and point 6 is the exterior point.)
This run calculates the fixed-boundary modes of each superelement, thus the output includes the eigenvalue summary table for each. Because this run has requests for only one mode per superelement, each superelement eigenvalue summary table has only one mode. The residual structure modes are improved over the static reduction case, but the modes are still not as accurate as we would like. Obtaining more modes for each superelement would improve the accuracy of the residual structure modes.

<table>
<thead>
<tr>
<th>MODE NO.</th>
<th>ORDER</th>
<th>EXTR</th>
<th>EIGVAL</th>
<th>CYCLES</th>
<th>MASS</th>
<th>STIFFNESS</th>
<th>GENERALIZED MASS</th>
<th>GENERALIZED STIFFNESS</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>7.518384E+07</td>
<td>8.670861E+03</td>
<td>1.380010E+03</td>
<td>1.000000E+00</td>
<td>7.518384E+07</td>
<td></td>
</tr>
</tbody>
</table>

**File seg10-c1 - Free-Free CMS of the Superelements**

The next example shows how to use free-boundary CMS for a superelement. Note that the exterior points are not constrained during the calculation of the component modes. Any SPCs on interior points are included, and if there are any constraints on the exterior points, these constraints are applied physically at the downstream superelement that has them as interior points, not at the current superelement.

We will use the SECSET1 entry to specify that we want free-free CMS. This entry has an option to state that ALL exterior DOFs will be unconstrained during the calculation of the component modes; that option is displayed here. If only selected DOFs were set free during the calculation of the component modes, they would be listed on SECSET and/or SECSET1 entries.
Once again, in this example only one mode per superelement is requested. More modes are required for greater accuracy.

ID seg10-cl
$ sample for cms - free boundary modes on a cantilever beam
$ using only one mode per superelement
SOL SEMODES
TIME 10000
CEND
TITLE = SEG10-1B - CANTILEVER BEAM DIVIDED INTO TWO SUPERELEMENTS
subtitle = free boundary cms - one mode per superelement
ECHO = NONE
DISPLACEMENT(PLOT) = ALL
SPC = 1
SUBCASE 1
LABEL = SUPERELEMENT 1
SUPER = 1
METHOD = 1
SUBCASE 2
LABEL = SUPERELEMENT 2
SUPER = 2
METHOD = 2
SUBCASE 3
LABEL = RESIDUAL STRUCTURE
SET 99 = 0
SUPER = 99
METHOD = 3
BEGIN BULK
$ SE 1 dynamic reduction
$ secset1,1,,all
EIGRL,1,,,1,0
SPOINT,1001,thru,1010
SEQSET1,1,0,1001,thru,1010
$
$ SE 2 dynamic reduction
$ secset1,2,,all
EIGRL,2,,,1,0
spoint,2001,thru,2010
seqset1,2,0,2001,thru,2010
$
$ residual structure
$
EIGRL,3,,,10,0
$
include ‘cantbeam.dat’
ENDDATA

The results of the above run follow. The input file requests the calculation of only one mode per superelement. The resulting eigenvalue summary tables appear below. Superelement 1 is the tip of the beam and (when run with all exterior DOFs in the C-set) has rigid-body modes; the one mode found is one of those rigid-body modes. The filtering process finds that this rigid-body mode is a linear combination of the static transformation vectors; this mode will be discarded. A better approach would be to list a lowest frequency of just above 0.0 hz on the EIGRL. Using this method, rigid-body modes would not be calculated.

Superelement 2 is the root of the beam, and interior point 1 is the point of fixity for the cantilever beam. Therefore, the expression free-free CMS does not really describe this superelement. The exterior grid point (GRID 6) is not constrained during the eigenvalue solution. However, interior grid point 1 has SPCs applied and is therefore constrained during the calculation of the modes. Thus, the component modes found for superelement 2 are the cantilevered modes of the component, which is
shown in the eigenvalue summary table for superelement 2. The first mode has a natural frequency of 1,380 Hz. Obviously, this mode is not a rigid-body mode and will be retained.

Notice that the frequencies are improved over the static reduction, but these frequencies are not as good as the fixed-boundary CMS results. There is no magic rule stating that free-free or fixed-boundary CMS are the best approaches for any model. You must use engineering judgement to determine which approach will provide component modes that best approximate the behavior of the superelement when it is attached to the rest of the model.

File seg10_d1.dat - Mixed Boundary CMS

Now we will demonstrate that mixed-boundary CMS. Superelement 1 is cantilevered from exterior grid point 6, thus fixed-boundary CMS is appropriate. That is, cantilevered modes represent how we expect this component to behave when it is attached to the rest of the structure. Superelement 2 also has exterior point 6; however, now it makes more sense to perform CMS with grid point 6 unconstrained. Fixed-boundary CMS does not represent how we expect this component to behave when it is attached to the rest of the model. Calculating the component modes with point 6 free (or cantilevered from interior grid point 1) gives a better representation. Thus, grid point 6 is placed in the C-set for superelement 2. These modes look more like we expect this component to behave when it is attached to the rest of the model.

```
ID seg10-d1
$ sample for cms - mixed boundary modes on a cantilever beam
$ using only one mode per superelement
SOL SEMODES
TIME 10000
CEND
TITLE = SEG10-1B - CANTILEVER BEAM DIVIDED INTO TWO SUPERELEMENTS
subtitle = mixed boundary cms -one modes per superelement
ECHO = NONE
DISPLACEMENT(PLOT) = ALL
SPC = 1
```
These results (again, with only one component mode per superelement) are an improvement over free-free CMS for this model, demonstrating the importance of selecting the proper treatment of the exterior points during CMS.

<table>
<thead>
<tr>
<th>MODE EXTRAC...</th>
</tr>
</thead>
<tbody>
<tr>
<td>NO.</td>
</tr>
<tr>
<td>1</td>
</tr>
<tr>
<td>2</td>
</tr>
<tr>
<td>3</td>
</tr>
<tr>
<td>4</td>
</tr>
<tr>
<td>5</td>
</tr>
<tr>
<td>6</td>
</tr>
<tr>
<td>7</td>
</tr>
<tr>
<td>8</td>
</tr>
</tbody>
</table>
Bulk Data Entries for Single-Level Dynamic Reduction of PARTs

To perform a dynamic reduction on a PART superelement, in addition to the required case control commands, you need to define the DOFs that will represent the dynamic transformation vectors (component modes). You also need to define the treatment of the exterior DOFs during the eigenvalue solution. (The default is fixed-boundary.)

When PARTs are used, their component modes can be represented using SPOINTs with QSET (or QSET1) entries in the Partitioned Bulk Data Section, or the SENQSET entry can be used in the Main Bulk Data Section to define how many (internal) DOFs will be allocated to the Q-set for each PART.

Once again, in a single-level dynamic solution all of the superelements attach directly to the residual structure. Therefore, component modes are passed to the residual structure for the system solution. In multilevel dynamics, we can create assemblies, where component modes from upstream superelements are coupled into a superelement; thus, we get the assembly modes before the reduced matrices are passed to the residual structure.

The following bulk data entries are used when performing dynamic reduction of PART superelements. All of these entries (except the SENQSET) must be in the Partitioned Bulk Data Section for the superelement.

**Note**

If you want a dynamic reduction of PART superelements, a Q-set must be defined for that PART. Use QSET or QSET1 entries pointing to SPOINTs in the partitioned bulk data or the SENQSET entry in the Main Bulk Data Section to define the Q-set.

**QSET and/or QSET1**

One or both of these entries can be used to define the DOFs used to represent the superelement dynamic transformation vectors (component modes) in subsequent processing; these entries point to SPOINTs. When PARTs exist in a model, grid points cannot be used for the Q-set. (Note: If you use the SENQSET, this entry is not required.)

**BSET and/or BSET1**

List of exterior DOFs to be constrained during the calculation of the dynamic transformation. (Default = all exterior DOFs.) Any DOFs listed on a BSET or BSET1 entry become exterior DOFs even if not selected by a SECONCT entry, automatic attachment, or ASET and/or ASET1 entries.

**Note**

For PARTs superelements, for each DOF listed on SUPORT entries, one dynamic transformation vector (starting with the lowest natural frequency) is removed. Because the program does not perform checking, we do not recommend this approach.

**CSET and/or CSET1**

List of exterior DOFs not constrained during the calculation of the dynamic transformation vectors. (Default = none.) DOFs listed on a CSET or CSET1 entry become exterior DOFs, even if not selected by a SECONCT entry, automatic attachment, or ASET and/or ASET1 entries.
SUPORT List of exterior DOFs to be placed into the superelement R-set. These DOFs are not constrained during the calculation of the dynamic transformation vectors; however, for each DOF listed on the SUPORT entry, one dynamic transformation vector is removed without any checking being performed.

SENQSET List of the number of automatically generated Q-set DOFs for selected superelements or all superelements. This entry can be used only if there are one or more PARTs in the deck. If used, this entry automatically generates internal DOFs for the Q-set for the selected superelements. These DOFs are passed to the residual structure with no coupling to downstream superelements until the residual structure is processed; therefore, this entry may not be the best choice for multilevel dynamic analysis.

SPOINT You must define the degrees of freedom NX Nastran uses for the component modes. (Note: If you use the SENQSET, this entry is not required.)

For PART superelements, component modes must be attached to SPOINTS. (An exception to this rule does exist—you can use the SENQSET entry to create internal DOFs to represent the dynamic vectors.) If SENQSET entries are used, internal DOFs are created and show up in the Superelement Connectivity List. These internal DOFs have a number range well above the user-accessible range of grid or SPOINT IDs. That is, the DOFs representing the dynamic vectors exist and are included in the solution, but you are unable to exercise any control over their use in the solution. To exercise control over the DOFs representing the dynamic vectors (for example, constraining a DOF to remove an unwanted vector), you must physically define the SPOINTs in the Partitioned Bulk Data Section for the PARTs, and you must have SPOINTs in the Main Bulk Data Section that are connected (by using SECONCT entries) to the SPOINTs representing the upstream modes. (See “Multilevel Dynamic Reduction” for an example.)

To attach the component modes to SPOINTs, use the QSET and/or QSET1 entries in the Partitioned Bulk Data Section for the PART with which you are dealing. When using these entries, select DOF 0 for the SPOINTs (which you have provided in the Partitioned Bulk Data Section) that represent the component modes. Make sure you have selected a sufficient number of DOFs to represent all of the component modes you are calculating, or the unrepresented modes will not be included in the solution. The SPOINTs that represent the component modes are exterior to the superelement. The next section on multilevel superelements in dynamics shows the method for having SPOINTs represent the component modes at the residual level.

Samples for Single-Level Dynamic Reduction Using PARTs

For single-level dynamic reduction using PARTs, we use the same model as before. However, now we define the two superelements using PARTs. Once again, this model is the cantilever beam used to generate the table comparing results of different reduction methods in “Introduction to Superelements in Dynamic Analysis”. In that chapter, several runs were made for each dynamic reduction approach, all of which were performed using SOL 103 (SEMODES).

The approaches we use are:

• Static reduction only (seg10p_a.dat)
• Fixed-boundary CMS (seg10p_b1.dat)
• Free-free CMS (seg10p_c1.dat)
• Mixed-boundary CMS (seg10p_d1.dat)

The following listings are the model used for this example. This model is the same cantilever beam shown in Figure 9-1 and is divided into two PARTs (1 and 2). Both PARTs have grid point 6 as an exterior point.

Note that both files are self-contained models. When each of these files is entered in a BEGIN SUPER section, the program determines that grid point 6 in PART 1 is coincident with grid point 6 from PART 2 and makes the connection. Remember, grid points are not required to have the same ID, but they must be coincident. For this model the same ID is used in both PARTs, but a different ID could have been used in each PART.

File cantp1.dat - Bulk Data for PART 1

```plaintext
$ file cantp1.dat
$ PARAM,GRDPNT,0
PBAR 1 1 0.1 0.01 0.01 0.001 0. +PR 1
+PR 1 0. 0. 0. 0. 0. 0. 0. +PA 1
+PA 1 0.
MAT1 1 2.9E+7 1.1E+7 0.327.357E-4 6.E-6 70. +MT 1
+MT 1
MAT4 15.787E-5 0.1167.357E-4
GRID 6 0 7.0. 0. 0.
GRID 7 0 6. 0. 0. 0.
GRID 8 0 7. 0. 0. 0.
GRID 9 0 8. 0. 0. 0.
GRID 10 0 9. 0. 0. 0.
GRID 11 0 10. 0. 0. 0.
CBAR 6 1 6. 7. 0. 0. 0.
CBAR 7 1 7. 8. 0. 0. 0.
CBAR 8 1 8. 9. 0. 0. 0.
CBAR 9 1 9. 0. 1. 0.
CBAR 10 1 10. 11. 0. 0.
```

File cantp2.dat - Bulk Data for PART 2

```plaintext
$ file cantp2.dat
$ PARAM,GRDPNT,0
SPC 1 1 123456 0.
PBAR 1 1 0.1 0.01 0.01 0.001 0. +PR 1
+PR 1 0. 0. 0. 0. 0. 0. 0. +PA 1
+PA 1 0.
MAT1 1 2.9E+7 1.1E+7 0.327.357E-4 6.E-6 70. +MT 1
+MT 1
MAT4 15.787E-5 0.1167.357E-4
GRID 6 0 7.0. 0. 0.
GRID 2 0 1. 0. 0. 0.
GRID 3 0 2. 0. 0. 0.
GRID 4 0 3. 0. 0. 0.
GRID 5 0 4. 0. 0. 0.
GRID 6 0 5. 0. 0. 0.
```
File seg10p_a.dat - Static Reduction Using PARTs

This file is similar to input file seg10_a.dat, except that we are using PARTs. Once again, the case control has a METHOD command that applies to all superelements (the default for the SUPER command is SUPER = ALL); however, because there are no DOFs provided in the superelement Q-sets to represent the component modes, the program performs a static condensation (Guyan reduction) only.

ID seg10p-a/sample
SOL SEMODES
TIME 10000
CEND
TITLE = seg10p-a - CANTILEVER BEAM DIVIDED INTO TWO PARTS
SUBTITLE = STATIC REDUCTION
METHOD = 1 $ note that since there are no qset entries, static
$ reduction is used for the superelements
ECHO = NONE
DISPLACEMENT(PLOT) = ALL
SPCFORCE(PLOT) = ALL
SPC = 1
BEGIN BULK
EIGRL 1 10 0
GRID 6 0 5. 0. 0. 0
begin super=1
include 'cantp1.dat'
$
begin super=2
include 'cantp2.dat'
ENDDATA

This run gives the same modes as seg10_a.dat. The boundary information for this run follows.

Note

No list of interior points is provided for the PARTs, only a list of the connecting (exterior) points. However, the residual structure (if there is a physical model) is defined in the Main Bulk Data Section and provides a list of interior points.

The following output shows that boundary point 1 (1B) is represented by grid point 6 in the residual structure, grid point 6 in PART 1, and grid point 6 in PART 2.

Note

The entry for grid point 6 in the Main Bulk Data is not necessary.

The list of exterior points for PARTs 1 and 2 identifies boundary point 1 (1B) as the exterior point, and the list of interior points for superelement 0 identifies boundary point 1 (1B) as an interior point.
File seg10p_b1.dat - Fixed Boundary CMS Using PARTs

This file is similar to file seg10_b1.dat, except that we are using PARTs. The case control is identical for the two files. The difference is that the superelements (PARTs) and the superelement Q-set DOFs are defined in the Partitioned Bulk Data Section for each superelement. The Q-set DOFs are defined by using SPOINTs and QSET1 entries. Notice that there are no SPOINT entries in the Main Bulk Data Section, and there are no SECONCT entries; therefore (at the residual structure), the DOFs representing the component modes from the superelements are internal DOFs and cannot be accessed by the user. (Usually, you do not need to access these points; however, you may want to constrain selected DOFs from the superelement Q-set at a collector or residual superelement to remove unwanted component modes.)

```plaintext
ID seg10p-b1, sample
$ sample for cms - fixed boundary modes on a cantilever beam
$ using only one mode per superelement
SOL SEMODES
TIME 10000
CEND
```
Once again, this is the same structure as that used in example seg10-b1, which results in the same modes; however, the components are now defined as PARTs and are automatically connected to each other. Also, there is no physical residual structure model; rather, the residual structure consists of internal DOFs created to represent the DOFs passed down from the upstream superelements.

The following example shows the boundary point lists. Notice that the SPOINTs used for the modes are identified as 1B through 10B (superelement 1) and 12B through 21B (superelement 2). Boundary point 11B is used to identify exterior grid point 6 in superelements 1 and 2. Because no equivalent grid point exists in the residual structure, boundary point 11B is represented by internal DOFs that are not accessible to the user. The following example shows the boundary point lists. Notice that the SPOINTs used for the modes are identified as 1B through 10B (superelement 1) and 12B through 21B (superelement 2). Boundary point 11B is used to identify exterior grid point 6 in superelements 1 and 2. Because no equivalent grid point exists in the residual structure, boundary point 11B is represented by internal DOFs that are not accessible to the user. These DOF are, however, included in the solution. No model is provided for the residual structure in this example; the residual structure consists only of copies of the boundary points from the upstream superelements and their component modes.
SEG10P-B1 - CANTILEVER BEAM DIVIDED INTO TWO PARTS  APRIL 29, 2004 NX Nastran 04/29/04
FIXED BOUNDARY CMS -ONE MODE PER SUPERELEMENT

BOUNDARY SEQUENCE ASSIGNMENT TABLE

<table>
<thead>
<tr>
<th>SEQUENCE</th>
<th>ASSIGNED TO POINT ID (SUPERELEMENT)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1B</td>
<td>1001</td>
</tr>
<tr>
<td>2B</td>
<td>1002 (1)</td>
</tr>
<tr>
<td>3B</td>
<td>1003 (1)</td>
</tr>
<tr>
<td>4B</td>
<td>1004 (1)</td>
</tr>
<tr>
<td>5B</td>
<td>1005 (1)</td>
</tr>
<tr>
<td>6B</td>
<td>1006 (1)</td>
</tr>
<tr>
<td>7B</td>
<td>1007 (1)</td>
</tr>
<tr>
<td>8B</td>
<td>1008 (1)</td>
</tr>
<tr>
<td>9B</td>
<td>1009 (1)</td>
</tr>
<tr>
<td>10B</td>
<td>1010 (1)</td>
</tr>
<tr>
<td>11B</td>
<td>6 (1)</td>
</tr>
<tr>
<td>12B</td>
<td>2001 (2)</td>
</tr>
<tr>
<td>13B</td>
<td>2002 (2)</td>
</tr>
<tr>
<td>14B</td>
<td>2003 (2)</td>
</tr>
<tr>
<td>15B</td>
<td>2004 (2)</td>
</tr>
<tr>
<td>16B</td>
<td>2005 (2)</td>
</tr>
<tr>
<td>17B</td>
<td>2006 (2)</td>
</tr>
<tr>
<td>18B</td>
<td>2007 (2)</td>
</tr>
<tr>
<td>19B</td>
<td>2008 (2)</td>
</tr>
<tr>
<td>20B</td>
<td>2009 (2)</td>
</tr>
<tr>
<td>21B</td>
<td>2010 (2)</td>
</tr>
</tbody>
</table>

SEG10P-B1 - CANTILEVER BEAM DIVIDED INTO TWO PARTS  APRIL 29, 2004 NX Nastran 04/29/04 \ PAGE 6
FIXED BOUNDARY CMS -ONE MODE PER SUPERELEMENT

POINT/SUPERELEMENT CONNECTIVITY LIST (SORTED BY POINT ID - "B" INDICATES BOUNDARY SEQUENCE ID)

<table>
<thead>
<tr>
<th>POINT ID</th>
<th>TYPE</th>
<th>SUPERELEMENT</th>
<th>CONNECTED TO SUPERELEMENT</th>
</tr>
</thead>
<tbody>
<tr>
<td>1B</td>
<td>S</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>2B</td>
<td>S</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>3B</td>
<td>S</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>4B</td>
<td>S</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>5B</td>
<td>S</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>6B</td>
<td>S</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>7B</td>
<td>S</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>8B</td>
<td>S</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>9B</td>
<td>S</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>10B</td>
<td>S</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>12B</td>
<td>S</td>
<td>0</td>
<td>2</td>
</tr>
<tr>
<td>13B</td>
<td>S</td>
<td>0</td>
<td>2</td>
</tr>
<tr>
<td>14B</td>
<td>S</td>
<td>0</td>
<td>2</td>
</tr>
<tr>
<td>15B</td>
<td>S</td>
<td>0</td>
<td>2</td>
</tr>
<tr>
<td>16B</td>
<td>S</td>
<td>0</td>
<td>2</td>
</tr>
<tr>
<td>17B</td>
<td>S</td>
<td>0</td>
<td>2</td>
</tr>
<tr>
<td>18B</td>
<td>S</td>
<td>0</td>
<td>2</td>
</tr>
<tr>
<td>19B</td>
<td>S</td>
<td>0</td>
<td>2</td>
</tr>
<tr>
<td>20B</td>
<td>S</td>
<td>0</td>
<td>2</td>
</tr>
<tr>
<td>21B</td>
<td>S</td>
<td>0</td>
<td>2</td>
</tr>
</tbody>
</table>

SEG10P-B1 - CANTILEVER BEAM DIVIDED INTO TWO PARTS  APRIL 29, 2004 NX Nastran 04/29/04 PAGE 10
FIXED BOUNDARY CMS -ONE MODE PER SUPERELEMENT

SUPERELEMENT 0

TYPE = RESIDUAL STRUCTURE

<table>
<thead>
<tr>
<th>INDEX</th>
<th>1B</th>
<th>2B</th>
<th>3B</th>
<th>4B</th>
<th>5B</th>
<th>6B</th>
<th>7B</th>
<th>8B</th>
<th>9B</th>
<th>10B</th>
</tr>
</thead>
<tbody>
<tr>
<td>LIST OF INTERIOR POINTS</td>
<td>12-0-0-0-0-0-0-0-0-0-0-1</td>
<td>(TOTAL NO. OF INTERIOR POINT = 21)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

SUPERELEMENT 1

TYPE = PRIMARY (BEGIN BULK SUPER)

<table>
<thead>
<tr>
<th>INDEX</th>
<th>1B</th>
<th>2B</th>
<th>3B</th>
<th>4B</th>
<th>5B</th>
<th>6B</th>
<th>7B</th>
<th>8B</th>
<th>9B</th>
<th>10B</th>
</tr>
</thead>
<tbody>
<tr>
<td>LIST OF EXTERIOR POINTS</td>
<td>12-0-0-0-0-0-0-0-0-0-0-1</td>
<td>(TOTAL NO. OF EXTERIOR POINT = 11)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

SUPERELEMENT 2

TYPE = PRIMARY (BEGIN BULK SUPER)

<table>
<thead>
<tr>
<th>INDEX</th>
<th>1B</th>
<th>2B</th>
<th>3B</th>
<th>4B</th>
<th>5B</th>
<th>6B</th>
<th>7B</th>
<th>8B</th>
<th>9B</th>
<th>10B</th>
</tr>
</thead>
<tbody>
<tr>
<td>LIST OF EXTERIOR POINTS</td>
<td>12-0-0-0-0-0-0-0-0-0-0-1</td>
<td>(TOTAL NO. OF EXTERIOR POINT = 11)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
File seg10p_c1.dat - Free-Free CMS Using PARTs

This file is similar to file seg10_c1.dat, except that we are now using PARTs. The case control is identical between the two files. The Q-set DOFs are defined by using an SENQSET entry in the Main Bulk Data Section. Therefore, the DOFs representing the component modes from the superelements are internal DOFs and can not be accessed by the user.

To perform free-free CMS, we use CSET1 (you can also use CSET entries) entries in the Partitioned Bulk Data Section for each PART. Any exterior DOFs that we do not want to constrain during the calculation of the component modes must be listed on CSET, CSET1, or SUPORT entries in the Partitioned Bulk Data Section for that PART.

Because we have asked the program to calculate the 0.0 hz modes of each PART, we are depending on the logic in the program to eliminate these modes. We could use SUPORT entries on selected exterior DOFs for the PARTs. For each exterior DOF defined on a SUPORT entry for a PART, one mode of the component is discarded, which is similar to the SESUP entry for main bulk data superelements. When calculating the modes of the PARTs, any exterior DOFs that are listed on CSET, CSET1, or SUPORT entries will not be constrained. Because there is a CSET1 entry in each Partitioned Bulk Data Section, the PARTs component modes are calculated with grid point 6 unconstrained, thus providing free-free modes for PART 2 and cantilevered modes for PART 1.

This sample demonstrates the use of the SENQSET entry. In the Main Bulk Data Section, we have asked that 10 internal DOFs be created to represent the component modes of each superelement in this model.

```
ID seg10p-c1
$ sample for cms - free boundary modes on a cantilever beam
$ using only one mode per superelement
SOL SEMODES
CEND
TITLE = SEG10P-C1 - CANTILEVER BEAM DIVIDED INTO TWO PARTS
subtitle = free boundary cms - one mode per superelement
ECHO = NONE
DISPLACEMENT(PLOT) = ALL
SPC = 1
SUBCASE 1
LABEL = SUPELEMENT 1
SUPER = 1
METHOD = 1
SUBCASE 2
LABEL = SUPELEMENT 2
SUPER = 2
METHOD = 2
SUBCASE 3
LABEL = RESIDUAL STRUCTURE
SET 99 = 0
SUPER = 99
METHOD = 3
BEGIN BULK
EIGRL,3,,,10,0
$
$ show use of SENQSET
$ SENQSET,all,10
$
begin super = 1
include ‘cantpl.dat’
$
$ SE 1 dynamic reduction
$```
The results of this run duplicate those in example seg10-c1. As in the previous example, there is no physical residual structure model. Instead, the residual structure consists of internal DOFs created to represent the DOFs passed down from the upstream superelements.

The following output shows the boundary point lists. The Boundary Sequence Assignment Table (Figure 10-2) finds grid point 6 in both superelements 1 and 2 coincident and identifies these points as point 1B. The next output shows all connecting points for the superelements. Notice that the internal DOFs created by the SENQSET entry are identified as 147477001B through 147477010B for superelement 2 and 147480001B through 147480010B for superelement 1. These numbers are outside of the range for defining grid or SPOINT entries and are not accessible to any input entries in NX Nastran. These numbers are internal DOFs generated solely to represent the component modes of the two superelements.
### Chapter 10: Input and Output for Dynamic Reduction

**Superelement User's Guide**

---

**SB10F-C1 - CANTILEVER BEAM DIVIDED INTO TWO PARTS**

**FREE BOUNDARY CMS - ONE MODE PER SUPERELEMENT**

#### BOUNDARY

**SEQUENCE ID: 10**

- **ASSIGNED TO POINT ID (SUPERELEMENT):**
  - 15

<table>
<thead>
<tr>
<th>POINT/SUPERELEMENT CONNECTIVITY LIST (SORTED BY POINT ID - &quot;B&quot; INDICATES BOUNDARY SEQUENCE ID)</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>INTERIOR TO</strong></td>
</tr>
<tr>
<td>POINT ID</td>
</tr>
<tr>
<td>1B</td>
</tr>
<tr>
<td>147477801B</td>
</tr>
<tr>
<td>147477802B</td>
</tr>
<tr>
<td>147477803B</td>
</tr>
<tr>
<td>147477804B</td>
</tr>
<tr>
<td>147477805B</td>
</tr>
<tr>
<td>147477806B</td>
</tr>
<tr>
<td>147477807B</td>
</tr>
<tr>
<td>147477808B</td>
</tr>
<tr>
<td>147477809B</td>
</tr>
<tr>
<td>147477810B</td>
</tr>
<tr>
<td>147480001B</td>
</tr>
<tr>
<td>147480002B</td>
</tr>
<tr>
<td>147480003B</td>
</tr>
<tr>
<td>147480004B</td>
</tr>
<tr>
<td>147480005B</td>
</tr>
<tr>
<td>147480006B</td>
</tr>
<tr>
<td>147480007B</td>
</tr>
<tr>
<td>147480008B</td>
</tr>
<tr>
<td>147480009B</td>
</tr>
<tr>
<td>147480010B</td>
</tr>
</tbody>
</table>

---

**SB10F-C1 - CANTILEVER BEAM DIVIDED INTO TWO PARTS**

**FREE BOUNDARY CMS - ONE MODE PER SUPERELEMENT**

#### COUNT POINT ID TYPE SUPERELEMENT CONNECTED TO SUPERELEMENT

<table>
<thead>
<tr>
<th><strong>INTERIOR TO</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>COUNT</td>
</tr>
<tr>
<td>1</td>
</tr>
<tr>
<td>10</td>
</tr>
</tbody>
</table>

---

**SB10F-C1 - CANTILEVER BEAM DIVIDED INTO TWO PARTS**

**FREE BOUNDARY CMS - ONE MODE PER SUPERELEMENT**

#### COUNT POINT ID TYPE SUPERELEMENT CONNECTED TO SUPERELEMENT

<table>
<thead>
<tr>
<th><strong>INTERIOR TO</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>COUNT</td>
</tr>
<tr>
<td>1</td>
</tr>
<tr>
<td>10</td>
</tr>
</tbody>
</table>

---

**SB10F-C1 - CANTILEVER BEAM DIVIDED INTO TWO PARTS**

**FREE BOUNDARY CMS - ONE MODE PER SUPERELEMENT**

#### COUNT POINT ID TYPE SUPERELEMENT CONNECTED TO SUPERELEMENT

<table>
<thead>
<tr>
<th><strong>INTERIOR TO</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>COUNT</td>
</tr>
<tr>
<td>1</td>
</tr>
<tr>
<td>10</td>
</tr>
</tbody>
</table>

---
File seg10p_d1.dat - Mixed-Boundary CMS Using PARTs

This file is similar to seg10_d1.dat, but we are using PARTs to define the model instead of main bulk data superelements. Superelement 2 is reduced using the default (fixed-boundary) CMS method, which treats superelement 2 as a cantilevered beam from grid point 6. Using the CSET1 entry for superelement 2, the component modes can be calculated with point 6 unconstrained or as a cantilever from point 1.

```
ID seg10p-d1
$ sample for cms - mixed boundary modes on a cantilever beam
$ using only one mode per superelement
SOL SEMODES
TIME 10000
CEND
    TITLE = SEG10p-d1 - CANTILEVER BEAM DIVIDED INTO TWO PARTS
    subtitle = mixed boundary cms -one modes per superelement
    ECHO = NONE
    DISPLACEMENT(PLOT) = ALL
    SPC = 1
    SUBCASE 1
    LABEL = SUPERELEMENT 1 - grid point 6 free during cms
    SUPER = 1
    METHOD = 1
    SUBCASE 2
    LABEL = SUPERELEMENT 2 - fixed boundary cms
    SUPER = 2
    METHOD = 2
    SUBCASE 3
    LABEL = RESIDUAL STRUCTURE
    SET 99 = 0
    SUPER = 99
    METHOD = 3
    BEGIN BULK
    EIGRL,3,,10,0
    senqset,all,10
    $
    begin super=1
    include 'cantpl.dat'
    $
```
$ SE 1 dynamic reduction
$ EIGRL,1,,,1,0
$ begin super=2
  include ‘cantp2.dat’
$ $ se 2 dynamic reduction
$ cset1,123456,6
  EIGRL,2,,,1,0
$ ENDDATA

The results of this problem are identical to the results for seg10-d1.

### 10.3 Multilevel Dynamic Reduction

Multilevel dynamic reduction is desirable for many models, though not required. As in static analysis, using a multilevel reduction reduces the size of the residual structure (resulting in a faster solution time for model change restarts). However, in dynamic analysis, multilevel reduction can be used to determine how assemblies perform before looking at the full structure. A good example of a structure on which you can use a multilevel reduction is your arm. Your hand cantilevers from the wrist, thus fixed-boundary modes of the hand behave as you expect them to. Then, looking at the forearm, it makes sense to create a collector superelement. The hand cantilevers off the forearm; therefore, you want to get assembly modes of the hand-forearm collector, cantilevered from the elbow. Once you have the assembly modes for the hand-forearm assembly, you can get modes of the complete arm assembly by attaching the hand-forearm collector to the upper arm and repeating the process. Each time, you are reducing the assembly to a single collector superelement with assembly modes.

Compare this method to single-level dynamic reduction, where the hand is reduced independently of the forearm, and both the hand and the forearm are reduced independently of the upper arm. Each component has its own modes, which do not include the effects of the other components until the residual structure, where assembly takes place. The component modes of each piece may behave differently when placed into the assembly; therefore, more component modes are required for each component to obtain the same accuracy as a multilevel reduction.

To perform multilevel dynamic reduction, you need to define the multilevel processing tree (DTI,SETREE or SETREE), which is mandatory if PARTs exist. You also need to manually define the DOFs that will be used for the component modes and the use of these DOFs in assemblies. As is the case with single level-reduction, the interface depends on how your superelements are defined.

#### Bulk Data Entries for Multilevel Dynamic Reduction

If your model has any PARTs, grid points cannot be used for the superelement Q-set; only SPOINTs can be used.

The interface for multilevel dynamic reduction for main bulk data superelements depends on whether the model has PARTs.

If your model has any PARTs, grid points cannot be used for the superelement Q-set; only SPOINTs can be used. When PARTs exist, SPOINTs can be interior to any superelement in your model.

If your model has no PARTs, then either grid points or SPOINTs can be used for the superelement Q-set. When there are no PARTs, SPOINTs (by definition) are interior to the residual structure;
however, the SPOINTS can be exterior to any superelement. Therefore, SPOINTS can be used to represent the component modes for any superelement. We recommend you use SPOINTS only for Main Bulk Data superelements that are immediately above the residual structure in the processing tree (at the first level above the residual structure). If SPOINTS are used to represent component modes of any superelements that occur higher in the processing tree, those component modes are passed directly to the residual structure and are not coupled into any downstream superelements until the final (residual structure) solution.

**Multilevel Dynamic Reduction for a Model With No PARTs**

This approach is identical to the approach used in versions prior to MSC.Nastran Version 69. The same entries are used as for a single-level dynamic reduction, but the component modes for the upstream superelements are represented by grid points. These grid points are placed interior to the superelement immediately downstream from the superelement using them for the Q-set. Thus, assembly modes are calculated and attached to the Q-set DOFs from the downstream (assembly) superelement.

The following example uses the multilevel version of the two-headed flyswatter model from “Examples of Multilevel Superelements”. Using multilevel dynamics, we request dynamic reduction for each superelement, and we use grid points to represent the component modes of the upstream superelements. We define these grid points as interior to the downstream superelements to get assembly modes.

For this example, the case control is created using the *expanded* format. Though not required, using this method affords total control over each superelement’s processing. Notice that in each subcase, we use the SUPER command to indicate the superelement with which we are working, and a METHOD command, to select the eigenvalue method for processing that superelement. A different EIGRL is not required for each superelement, but this specification will minimize effort in future runs if we are using restarts. If a separate subcase and eigenvalue method are specified for each superelement, you can make changes that are unique to a single superelement more easily. If we use the condensed format, a change to the single subcase affects all superelements, removing much of the efficiency associated with superelements when using restarts.

```
$ mult103_a.dat - multi-level dynamics - main bulk data only
$
SOL 103
CEND
TITLE = file multi1 -multi-level tree - main bulk data only
SUBTITLE = STATICS
DISP = ALL
$
$ SUBCASE 11
LABEL = DYNAMIC REDUCTION OF TIP SUPERELEMENT 1
SUPER = 1
METHOD = 11
$
$ SUBCASE 21
LABEL = DYNAMIC REDUCTION OF TIP SUPERELEMENT 2
SUPER = 2
METHOD = 21
$
$ SUBCASE 31
LABEL = DYNAMIC REDUCTION OF ASSEMBLY SUPERELEMENT 3
SUPER = 3
METHOD = 31
$
$ SUBCASE 41
```
LABEL = DYNAMIC REDUCTION OF ASSEMBLY SUPERELEMENT 4
SUPER = 4
METHOD = 41
$
$
SUBCASE 51
LABEL = DYNAMIC REDUCTION OF ASSEMBLY SUPERELEMENT 5
SUPER = 5
METHOD = 51
$
$
SUBCASE 61
LABEL = DYNAMIC REDUCTION OF ASSEMBLY SUPERELEMENT 6
SUPER = 6
METHOD = 61
$
$
SUBCASE 71
LABEL = DYNAMIC REDUCTION OF TIP SUPERELEMENT 7
SUPER = 7
METHOD = 71
$
$
SUBCASE 1001
LABEL = RESIDUAL STRUCTURE SOLUTION
set 99 = 0
SUPER = 99
METHOD = 101
$
$
BEGIN BULK
$
DTI,SETREE,0,2,4,4,6,6,0,
,1,3,3,5,5,0,7,0
include 'seset.mult'
INCLUDE 'model.dat'
INCLUDE 'load1.dat'
$
$ entries for multi-level dynamic reduction
$
GRID,1001
GRID,1002
SEQSET1,1,123456,1001,1002
EIGRL,11,,1000.
$
$
GRID,2001
GRID,2002
SEQSET1,2,123456,2001,2002
EIGRL,21,,1000.
$
$
$ put component modes from superelement 1 in superelement 3
$
$
SESET,3,1001,1002
GRID,3001
GRID,3002
GRID,3003
SEQSET1,3,123456,3001,3002,3003
EIGRL,31,,800.
$
$
$ put component modes from superelement 2 in superelement 4
$
$
SESET,4,2001,2002
GRID,4001
GRID,4002
GRID,4003
SEQSET1,4,123456,4001,4002,4003
EIGRL,41,,800.
$
$
$ put component modes from superelement 3 in superelement 5
$
$
SESET,5,3001,3002,3003
To understand how the multilevel process works, we will look at the branch of the tree containing superelements 1, 3, and 5, above the residual structure.

Looking at the bulk data, we see that the model definition and the multilevel tree are identical to a static run. The changes for dynamic multilevel processing are shown after the main model definition. By way of explanation, we will look at the input immediately following entries for multilevel dynamic reduction. This area contains the entries used to control the multilevel processing. The entries for superelement 1 are:

```
GRID,1001
GRID,1002
SEQSET1,1,123456,1001,1002
EIGRL,11,,1000.
```

We are defining two new grid points that have no connection to the model. (DO NOT CONNECT ANY STRUCTURAL ELEMENTS TO GRID POINTS USED IN A SUPERELEMENT Q-SET.) Both of these points are defined at the Basic Origin (they have no value in the location fields), which would not be allowed if the model had PARTs. However, because (at this point) we are only looking at models without PARTs, this method is acceptable. The SEQSET1 entry instructs NX Nastran to use all six DOFs of both of these two new points to represent the modes of superelement 1. Therefore, the first component mode is attached to DOF 1 of GRID 1001, the second mode is attached to DOF 2 of GRID 1001, and so on. SUBCASE 11 instructs NX Nastran to use EIGRL 11 to solve for modes of superelement 1 up to 1000.0 hz. Because there are no SECSETi entries, the modes are calculated with the exterior points (grid points 35 and 36) held fixed, or the component is cantilevered from the interface with superelement 3.

The following eigenvalue Summary Table (Figure 10-3) represents the fixed-boundary modes of superelement 1. There are six modes under the 1000.0 hz cutoff. These modes are attached to DOFs 1-6 of grid point 1001 and passed downstream. (Because there are only six modes, the DOFs of GRID 1002 do not have any data associated with them and will be removed by AUTOSPC in the downstream superelement.)
Figure 10-3. Eigenvalue Summary Table.

Using these component modes and the constraint modes, superelement 1 is reduced to matrices attached to its exterior DOFs.

Now that superelement 1 is completely processed, let us proceed down the processing tree. The superelement below superelement 1 is superelement 3. We attach the reduced matrices from superelement 1 to the finite element model of superelement 3, and we prepare to reduce superelement 3 to its exterior points. The following entries for superelement 3 are in the input file (beyond the standard superelement definition).

$  
$ put component modes from superelement 1 in superelement 3  
$  
SESET,3,1001,1002  
GRID,3001  
GRID,3002  
GRID,3003  
SEQSET1,3,123456,3001,3002,3003  
EIGRL,31,800.

The first of these entries instructs NX Nastran to place the grid points (1001 and 1002) that represent the modes of superelement 1 interior to superelement 3, allowing us to obtain the assembly modes. If we did not have this entry, the grids representing the modes from superelement 1 would be exterior to superelement 3 and would not couple into the problem until we processed the residual structure.

We now define some new grid points (3001, 3002, and 3003), which will represent the modes for the assembly. The SEQSET1 instructs NX Nastran to use these points to represent the assembly modes in a manner similar to the previous set (first mode = GRID 3001, DOF 1, and so on).

The EIGRL, selected by the METHOD in SUBCASE 31, instructs NX Nastran to solve for assembly modes up to 800.0 hz. Because we are working with the assembly of superelements 1 and 3, the assembly modes have lower frequencies than those for superelement 1 alone; therefore, we can use a lower cutoff frequency than we used for superelement 1 alone. Figure 10-4 shows the modes for assembly superelement 3.
Figure 10-4. The Grid Point Singularity Table

The Grid Point Singularity Table (Figure 10-4) indicates that all six DOFs for grid point 1002 are singular, which agrees with our earlier results. That is, the six modes for superelement 1 are attached to the six DOFs of grid point 1001, and no data is associated with the DOFs of GRID 1002.

The modes of the assembly are calculated with the exterior points fixed, meaning that for this assembly the program calculates the cantilevered modes for superelements 1 and 3 constrained at grid points 19 and 20. Assembly superelement 3 has seven modes under 800.0 hz. The first mode of the assembly is at 25.6 hz, compared to the first natural frequency of superelement 1, which is 81.6 hz. These modes are attached to DOFs 1-6 of grid point 3001 and DOF 1 of grid point 3002. (No data is associated with DOFs 2-6 of GRID 3002 or DOFs 1-6 of GRID 3003.)

The matrices for assembly superelement 3 are reduced to the exterior DOFs; then we proceed to superelement 5, which is the next level in the current branch.

We now attach the reduced matrices from assembly superelement 3 to the physical model of superelement 5 to form a new assembly. The following bulk data entries instruct NX Nastran how to accomplish this new assembly.

```
$  put component modes from superelement 3 in superelement 5
$  SESET,5,3001,3002,3003
GRID,5001
GRID,5002
GRID,5003
SEQSET1,5,123456,5001,THRU,5004
EIGRL,51,,600.
```

The SESET entry instructs the program to place grid points 3001, 3002, and 3003 (which we used to represent the component modes of superelement 3) interior to superelement 5. Once again, we are calculating assembly modes. We now define three additional grid points (5001, 5002, and 5003), which represent the modes of assembly superelement 5. Using EIGRL 51 (as instructed in SUBCASE 51), we solve for modes of the assembly up to 600.0 hz. Again, the frequency range is decreased as we go down the branch, because the assemblies have more modes in a lower range.
Now we solve the fixed-boundary eigenvalue problem. Assembly superelement 5 (containing the reduced information representing superelements 1 and 3) is solved for fixed-boundary modes with exterior points 13 and 23 constrained. Figure 10-5 shows the Assembly Eigenvalue Summary Table.

![Figure 10-5. Assembly Eigenvalue Summary Table.](image)

As expected, the Grid Point Singularity Table shows that DOFs 2-6 of grid point 3002 and DOFs 1-6 of grid point 3003 are singular and constrained. These are the unused Q-set DOFs from upstream superelement 3. (We calculated seven modes and attached them to DOFs 1-6 of GRID 3001 and DOF 1 of GRID 3002). The seven modes calculated for assembly superelement 5 are attached to DOFs 1-6 of grid point 5001 and DOF 1 of grid point 5002 and passed to the residual structure.

The branch containing superelements 2, 4, and 6 is processed in a similar manner. Superelement 7 is a tip superelement that connects directly to the residual structure and requires no special effort (beyond definition of the Q-set DOFs).

The assembly model is created at the residual structure, and the assembly modes are found. The following Eigenvalue Table (Figure 10-6) represents the modes for superelement 0.
Figure 10-6. Eigenvalue Table.

For comparison, we will run the model with no superelements. The following file (modesnose.dat) solves for the modes of the model under 500 hz without any reduction.

```
$ file - modesnose.dat - normal modes without superelements
SOL 103
CEND2001
TITLE = file modesnose - normal modes - no superelements
DISP = ALL
METHOD = 101
BEGIN BULK
INCLUDE 'model.dat'
ASSEMBLY MODES
EIGRL,101,,500.
ENDDATA
```

The following Eigenvalues Results Table (Figure 10-7) result from running file modesnose.dat:

```
Figure 10-7. Eigenvalue Results.

The difference in the frequency for the thirteenth mode (407hz) is .33%, indicating the CMS performed very well. Note that the fourteenth mode of the system occurs at 499.36 hz and does not show up in the results of the multilevel solution, due to round off (truncation) in the solution. If we extend the
frequency range beyond 500.0 hz, then this mode is found. Also note that round off is difficult (if not impossible) to predict; therefore, results should always be verified by using either another reduction (for example, additional component modes), a full-model solution, or a test.

Commonly in CMS, modes for the components should be found to at least 1 1/2 times the frequency range of interest. Had we followed this rule for assembly superelement 5, we would have performed the CMS to at least 750 hz, rather than the 600 hz cutoff used in the example, which might have been enough to allow the fourteenth mode to be found under 500 hz. Note that the recommendation that component modes be found to at least 1 1/2 times the frequency of interest does not guarantee accuracy, and has no foundation in theory. It is simply a common practice.

If a multilevel model contains PARTs, it is mandatory for the processing tree to be defined using either SETREE or DT1,SETREE entries.

**Bulk Data Entries for Multilevel Dynamic Reduction When PART Superelements Exist**

As mentioned in "Multilevel Superelement Analysis", if a model contains PARTs, NX Nastran automatically treats that model as a single-level model unless either SETREE or DT1,SETREE entries exist. Therefore, unlike a model defined using only main bulk data superelements, a model using PARTs requires the multilevel tree to be defined if multilevel processing is desired.

If PARTs exist in a model, only SPOINTs or the SENQSET can be used to define the Q-set for a superelement.

When PARTs exist, only SPOINTs or the SENQSET can be used to define the Q-set DOFs of a superelement. Unless you manually request the program to couple these modes into the downstream superelement, the modes are passed directly to the residual structure and will not be coupled into any downstream superelement component modes. Although not identical, this method is similar to the Benfield-Hruda method of CMS.

Using this default (passing modes directly to the residual and not including these modes in assembly component modes) is not as accurate as coupling a superelement's modes into the downstream superelement (including the effect of the superelement's modes on the assembly modes). Example problems can be created for any of the modern techniques, wherein one approach greatly outperforms all other approaches. However, coupling upstream superelements into assembly modes generally provides a greater degree of accuracy for a smaller number of DOFs at the final (residual structure) solution.

To prevent the program from automatically passing a superelement's modes all the way to the residual structure solution, you must manually put the DOFs representing those modes into the downstream superelement that you want to include these modes. Therefore, you must define additional SPOINTs, which are placed interior to the downstream superelement, and use the SECONCt entry to instruct the program to attach the upstream modes to these DOFs. Once the model contains a PART, you must repeat this process for both main bulk data superelements and PARTs if you want to get assembly modes.

Note that any Q-set DOFs defined using the SENQSET entry automatically belong to the residual structure. These DOFs cannot be combined into downstream superelement assemblies until the final (residual structure) solution.

The following input demonstrates how to couple the modes from an upstream superelement into a downstream superelement. We assume that superelement 1 is above superelement 3 in our superelement tree. We will perform CMS on superelement 1, but now we want to obtain assembly
modes when processing superelement 3; that is, we want to observe the behavior of the assembly of superelements 1 and 3. The following input accomplishes this task.

```
SOL 103
CEND
TITLE = ASSEMBLY MODES
DISP(PLOT)=ALL
$
SUBCASE 1
LABEL = CMS OF SUPERELEMENT 1
SUPER = 1
METHOD = 100
$
SUBCASE 2
LABEL = ASSEMBLY MODES OF SE 1 AND 3
SUPER = 3
METHOD = 101
$
BEGIN BULK
$
$ MAIN BULK DATA SECTION
$
$ ENTRIES TO “COUPLE” MODES FROM SUPERELEMENT 1 INTO ASSEMBLY SE 3
$
SETREE, 3, 1  $ place SE 1 above SE 3 in tree
$
$ "CONNECT" UPSTREAM MODES TO DOF INTERIOR TO SE 3
$
SECONCT, 1, 3
, 1001, 11001, 1002, 11002, 1003, 11003, 1004, 11004
, 1005, 11005, 1006, 11006, 1007, 11007, 1008, 11008
, 1009, 11009, 1010, 11010, 1011, 11011, 1012, 11012
$
$
BEGIN SUPER=1
$
$ DEFINE Q-SET FOR SE 1
$
SPOINT, 1001, THRU, 1012
QSET1, 0, 1001, THRU, 1012
EIGRL, 100,.....
$
$
BEGIN SUPER=3
$
$ DEFINE DOF TO ATTACH UPSTREAM MODES TO
$
SPOINT, 11001, THRU, 11012
$
$ DEFINE Q-SET FOR “ASSEMBLY” SE 3
$
SPOINT, 3001, THRU, 3015
QSET1, 0, 3001, THRU, 3015
EIGRL, 101,.....
$
ENDDATA
```

The case control in the above input should be self-explanatory. We are requesting CMS for superelements 1 and 3. In the Partitioned Bulk Data Section for superelement 1, we define SPOINTs 1001 through 1012. These SPOINTs are used as the DOFs to represent the component modes calculated using EIGRL number 100 (also in this section). Once superelement 1 is processed, we are ready to continue down the branch and begin processing superelement 3, which is our assembly. In the partitioned bulk data for superelement 3 we not only define SPOINTs 3001 through 3015, which
will represent the assembly modes, but we also define SPOINTs 11001 through 11012, to which
the modes from superelement 1 are connected.

**Note**

Because each Partitioned Bulk Data Section is treated separately, the same numbers can be
used for the SPOINTs as are used in the upstream superelement, which still requires the use
of a SECONCT entry to attach the modes. The upstream modes are attached to these DOFs
by the use of the SECONCT entry in the Main Bulk Data Section. This entry instructs the
program to connect the DOFs that represent the modes of superelement 1 to the new DOFs
we have defined interior to superelement 3.

**Sample of Multilevel CMS Using PARTs**

The following input file is similar to mult103-a.dat (shown earlier in this section), except that PARTs
are used to define the superelements.

```
$ use spoints
$ selm103p - multi-level using PARTs
ID SE, SAMPLE PROBLEM - MULTI-LEVEL DYNAMICS
SOL 103
CEND
$
TITLE = file -multi-level tree - PARTs
SUBTITLE = MODES
DISP = ALL
$
SUBCASE 11
LABEL = DYNAMIC REDUCTION OF TIP SUPERELEMENT 1
SUPER = 1
METHOD = 11
$
SUBCASE 21
LABEL = DYNAMIC REDUCTION OF TIP SUPERELEMENT 2
SUPER = 2
METHOD = 21
$
SUBCASE 31
LABEL = DYNAMIC REDUCTION OF ASSEMBLY SUPERELEMENT 3
SUPER = 3
METHOD = 31
$
SUBCASE 41
LABEL = DYNAMIC REDUCTION OF ASSEMBLY SUPERELEMENT 4
SUPER = 4
METHOD = 41
$
SUBCASE 51
LABEL = DYNAMIC REDUCTION OF ASSEMBLY SUPERELEMENT 5
SUPER = 5
METHOD = 51
$
SUBCASE 61
LABEL = DYNAMIC REDUCTION OF ASSEMBLY SUPERELEMENT 6
SUPER = 6
METHOD = 61
$
SUBCASE 71
LABEL = DYNAMIC REDUCTION OF TIP SUPERELEMENT 7
SUPER = 7
METHOD = 71
$
```

10-34 Superelement User's Guide
SUBCASE 1001
LABEL = RESIDUAL STRUCTURE SOLUTION
set 99 = 0
SUPER = 99
METHOD = 101
$ include 'plot.blk'
$ BEGIN BULK
$ setree,3,1
setree,5,3
setree,4,2
setree,6,4
setree,0,5,6,7
EIGRL,101,,500.
include 'part0.dat'
$ combine upstream Q-set dof in downstream superelements
seconct,1,3,
,1001,11001,1002,11002,1003,11003,1004,11004
,1005,11005,1006,11006,1007,11007,1008,11008
,1009,11009,1010,11010,1011,11011,1012,11012
seconct,2,4
seconct,3,5,
,3001,13001,3002,13002,3003,13003,3004,13004
,3005,13005,3006,13006,3007,13007,3008,13008
,3009,13009,3010,13010,3011,13011,3012,13012
,3013,13013,3014,13014,3015,13015
seconct,4,6
,4001,14001,4002,14002,4003,14003,4004,14004
,4005,14005,4006,14006,4007,14007,4008,14008
,4009,14009,4010,14010,4011,14011,4012,14012
,4013,14013,4014,14014,4015,14015
begin super=1
$ EIGRL,11,,1000.
spoint,1001,thru,1012
qset1,0,1001,thru,1012
include 'part1.dat'
$ begin super=2
EIGRL,21,,1000.
spoint,2001,thru,2012
qset1,0,2001,thru,2012
include 'part2.dat'
$ begin super=3
EIGRL,31,,800.
$ spoints to attach se 1 modes to
spoint,11001,thru,11012
spoint,3001,thru,3015
qset1,0,3001,thru,3015
include 'part3.dat'
$ begin super=4
EIGRL,41,,800.
spoint,12001,thru,12012
spoint,4001,thru,4015
qset1,0,4001,thru,4015
include 'part4.dat'
$ begin super=5
spoint,13001,thru,13015
spoint, 5001, thru, 5020
qset1, 0, 5001, thru, 5020
EIGRL, 51, , 600.
include 'part5.dat'
$
begin super=6
spoint, 14001, thru, 14015
spoint, 6001, thru, 6020
qset1, 0, 6001, thru, 6020
EIGRL, 61, , 600.
include 'part6.dat'
$
begin super=7
EIGRL, 71, , 1000.
spoint, 7001, thru, 7010
qset1, 0, 7001, thru, 7010
include 'part7.dat'
$
ENDDATA

The Executive and Case Control Sections of this file are identical to mult103_a.dat. The bulk data is for the same model, except that PARTs are used to define the model. The tree is identical to mult103_a.dat, and the answers are the same.

Notice that (as shown earlier) SECONCT entries are used in the Main Bulk Data Section to place the modes from upstream superelements into the downstream superelements. The use of SECONCT entries is comparable to employing grid entries and SESET entries when the model uses main bulk data superelements only.

The procedure shown in this sample applies to all multilevel models using PARTs, regardless of whether main bulk data superelements are defined. If a PART is defined, then this procedure must be used to couple the modes into downstream superelements, unless you wish all upstream superelement modes to be passed directly to the residual structure.
Chapter 11: Dynamic Loading on Superelements

- Case Control Used to Define Dynamic Loadings on Superelements using LOADSET -LSEQ
- Example Demonstrating Dynamic Loads on Superelements
When performing a dynamic analysis with loads on superelements, you can use DAREA entries. You cannot use DELAY or DPHASE entries for loads inside superelements. In previous versions, loads inside superelements could only be defined using the LOADSET Case Control command and the LSEQ Bulk Data entry.

Any loading applied on the residual structure can be defined in the same manner as for non-superelement models.

## 11.1 Case Control Used to Define Dynamic Loadings on Superelements using LOADSET -LSEQ

If you want, you can define dynamic loadings on superelements using the LOADSET Case Control command. Each superelement can use a different LOADSET command; all superelements can use a single LOADSET command; or you can use LOADSET commands for groups of superelements. The same rules apply to the case control as apply in static analysis. That is, you can use either expanded or condensed case control, depending on the problem you are solving.

The format of the LOADSET command is simple; it is

**LOADSET = i**

where i = the SID of one or more LSEQ entries in the Bulk Data Section of your file. For each superelement with a LOADSET command in the Case Control Section, NX Nastran generates and reduces loading vectors based on the LSEQ Bulk Data entries.

### Bulk Data

When using LOADSET -LSEQ, dynamic loads for superelements are defined by telling NX Nastran the static loading sets you want to apply dynamically, which is accomplished using the LSEQ entry. The format of the LSEQ entry is:

#### Format:

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
</tr>
</thead>
<tbody>
<tr>
<td>LSEQ</td>
<td>SID</td>
<td>EXCITEID</td>
<td>LID</td>
<td>TID</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

The LSEQ entry instructs NX Nastran to convert a static and/or temperature loading into an equivalent DAREA ID for dynamic application. Though a conversion does not actually occur in the program, it is perhaps the easiest way to think of this process.

The fields on this entry are used as follows:

- **SID**: Set ID for this LSEQ. Any number of LSEQ entries can use the same SID selected by the LOADSET command.

- **EXCITEID**: Set ID of the DAREA set this entry is defining. This is the ID used on the TLOADi or RLOADi entries to apply this loading dynamically. All loads with the same DAREA ID are combined at the residual structure.
LID  Set ID of the static loading to be converted into DAREA.

TID  Set ID of the temperature loading to be converted into DAREA.

Once again, it is easiest to think of this entry as instructing the program to internally convert a static loading into DAREA entries. What really happens when the LOADSET-LSEQ combination is used is described below.

When a LOADSET Case Control command is encountered while processing a superelement, the program finds all LSEQ entries that match the SID on the LOADSET command. Using these LSEQ entries, a column in the static loading matrix (PJ) is created for the current superelement for each EXCITEID set identified on these entries. The program then places the associated loading (LID and/or TID) data into that column of the loading matrix and processes that matrix for the superelement. The OLOAD RESULTANT output for the superelement is defined for the loadings in this manner, and each column corresponds with a EXCITEID set defined on the LSEQ entries, in ascending order. These loading matrices go through the SELG and SELR operations for all superelements in the model. At the residual structure, the reduced superelement loading matrices are combined with loadings defined for the residual structure to form a PG matrix that is associated with the EXCITEID sets selected. These loadings are then combined with DAREA entries (only allowed for the residual structure) to define the complete set of dynamic loadings to be applied (selected by TLOADi or RLOADi entries).

Once again, DAREA, DELAY and DPHASE entries can be used only for the residual structure. If you want to apply a delay or phase angle to a loading on a superelement, you must do so in the actual dynamic loading definition using the RLOADi or TLOADi and its associated data.

### 11.2 Example Demonstrating Dynamic Loads on Superelements

Using the model (cantbeam.dat) from “Introduction to Superelements in Dynamic Analysis”, we will look at defining dynamic loadings on superelements. For this example, we assume that we have a loading of two units applied in the Y-direction at grid point 11 of the model. This loading is applied dynamically in a transient solution and oscillates at 10 hz in a sine function.

![Diagram of superelements](image)

In this model, the cantilever beam is divided into two superelements and a residual structure. The residual structure contains grid point 6 and DOFs to represent the component modes of the
superelements. The default (Craig-Bampton)\(^1\) method is used for CMS, with three modes per superelement.

```
ID seg12-a1, sample
$ sample to demonstrate loadings on superelements in dynamics
$
SOL 112
CEND
TITLE = SEG11-A1 - CANTILEVER BEAM DIVIDED INTO TWO SUPERELEMENTS
subtitle = fixed boundary cms -three modes per superelement
label = transient response with load in superelement 2
ECHO = NONE
DISPLACEMENT(PLOT) = ALL
SPC = 1
$
LOADSET = 999
DLOAD = 101
TSTEP = 102
$
SUBCASE 1
  LABEL = SUPERELEMENT 1
  SUPER = 1
  METHOD = 1
SUBCASE 2
  LABEL = SUPERELEMENT 2
  SUPER = 2
  METHOD = 2
SUBCASE 3
  LABEL = RESIDUAL STRUCTURE
  SET 99 = 0
  SUPER = 99
  METHOD = 3
BEGIN BULK
$
$ dynamic loading definition
$
LSEQ,999,1010,10
FORCE,10,11,,2.,0.,1.,0.
TLOAD2,101,1010,,,0.,1000.,10.,-90.
TSTEP,102,100,.01
$
$ SE 1 dynamic reduction
$
EIGRL,1,,,3,0
SPOINT,1001,thru,1010
SEQSET1,1,0,1001,thru,1010
$
$ se 2 dynamic reduction
$
EIGRL,2,,,3,0
spoint,2001,thru,2010
seqset1,2,0,2001,thru,2010
$
$ residual structure
$
EIGRL,3,,,10,0
$
include 'cantbeam.dat'
ENDDATA
```

The above file uses SOL 112 (modal transient) to perform the solution. SOL 109 (direct transient) could have also been used. The solution selected is independent of the methodology used to reduce the superelements. The entries that are used to define the dynamic loading are:

```
LSEQ,999,1010,10
FORCE,10,11,,2.,0.,1.,0.
TLOAD2,101,1010,,0.,1000.,10.,-90.
TSTEP,102,100,,01.
```

Though not required, the LOADSET, DLOAD, and TSTEP commands are above the first SUBCASE in the case control. These commands can occur within the SUBCASES; however, the first approach is easier. The DLOAD and TSTEP commands are mandatory for the residual structure SUBCASE and optional for the superelements SUBCASES. If you are not requesting OLOAD output, you do not need to place these commands in the superelement SUBCASES. (The only purpose these commands serve for superelements is to request OLOAD output.) NX Nastran generates static loading vectors at the superelement level. The program then passes these vectors through the reduction process until the residual structure, where these static vectors are combined with associated DAREA entries, and the information from the DLOAD and TSTEP is used to solve the problem.

You can use a different LOADSET command for each superelement in the model, but it is easier to use a single LOADSET command and to place it above all SUBCASES. In this model, the LOADSET command instructs NX Nastran to use LSEQ set 999 to create static loading vectors that will be applied dynamically.

In the Bulk Data Section, the LSEQ, TLOAD, and TSTEP entries define the dynamic solution. The LSEQ entry (set 999) instructs the program to create a static loading vector based on static load set 10. This loading vector is associated with EXCITEID set 1010.

Static loading 10 is defined by the FORCE entry, which states that we have a 2.0 unit force in the positive Y-direction on grid point 11. This grid point is interior to superelement 1; therefore, when superelement 1 is processed, a loading vector is generated. During the processing, the output for superelement 1 includes OLOAD RESULTANT, as follows:

```
TSTEP,109,200,,01.
```

This output verifies that we have a 2.0 unit load in the plus Y-direction. This loading is reduced with the other matrices from superelement 1, and at the residual structure this loading is associated with DAREA set 1010, which the TLOAD2 applies dynamically. Compare this with the following residual-structure-only model, which uses DAREA entries.

```
ID seg12-a2, sample
$ reference sample for dynamic loads - DAREA
$ $ SOL 112
TIME 10000
CEND
TITLE = SEG11-A1 - CANTILEVER BEAM DIVIDED INTO TWO SUPERELEMENTS
subtitle = fixed boundary cms -three modes per superelement
label = transient response with load in superelement 2
```

In this file, the DAREA entry is used to define the dynamic loading. DAREA set 1010 defines a loading with a magnitude of 2.0 units, which is applied at grid point 11 in the Y-direction. TLOAD2 101 applies this loading dynamically.

For comparison, the following file is a residual-structure-only model that uses the LOADSET-LSEQ method to define the loadings. Aside from the lack of superelement entries and the CMS requests, this file is identical to file seg12_a1.dat (the model using superelements).

```
ECHO = NONE
DISPLACEMENT(PLOT) = ALL
SPC = 1
$
DLOAD = 101
TSTEP = 102
$
SUBCASE1
LABEL=SE1
METHOD=1
SUBCASE=2
LABEL=SE2
METHOD=2
SUBCASE3
SET 99 = 0
SUPER = 99
METHOD = 3
BEGIN BULK
param,post,0
$
$ dynamic loading definition
$ DAREA,1010,11,2,2.
TLOAD2,101,1010,,,0.,1000.,10.,-90.
TSTEP,102,100,.01
$
$ SE 1 dynamic reduction
$
EIGRL,1,,,3,0
SPOINT,1001,thru,1010
SEQSET1,1,0,1001,thru,1010
$
$ se 2 dynamic reduction
$
EIGRL,2,,,3,0
spoint,2001,thru,2010
seqset1,2,0,2001,thru,2010
$
$ residual structure
$
EIGRL,3,,,10,0
$
include 'cantbeam.dat'
.
.
ENDDATA
```

ID seg12-a2, sample
$ reference sample for dynamic loads - no superelements
$
SOL 112
TIME 10000
CEND

title = SEG11-A1 - CANTILEVER BEAM
label = transient response
ECHO = NONE
DISPLACEMENT(PLOT) = ALL
Dynamic Loading on Superelements

```
SPC = 1
$
LOADSET = 999
DLOAD = 101
TSTEP = 102
$
  SET 99 = 0
  SUPER = 99
  METHOD = 3
BEGIN BULK
  param,post,0
$
  $ dynamic loading definition
  $
  LSEQ,999,1010,10
  FORCE,10,11,,2.,0.,1.,0.
  TLOAD2,101,1010,,0.,1000.,10.,-90.
  TSTEP,102,100,.01
$
  EIGRL,3,,,10,0
$
$
$ the rest is identical to cantbeam.dat with the "seset" entries removed
$
$
ENDDATA
```

All three of these input files provide the correct solution for the problem; the choice of methods is yours.

**Dynamic Loadings Applied on PARTs**

Although the idea is the same, there is an additional requirement when using PARTs in your model. If you use a LOADSET command in the Case Control Section, you must have an associated LSEQ entry in each section of your partitioned input file that contains a superelement (or PART) to which the LOADSET command applies (even for PARTs that have no dynamic loading). If you fail to do this, you will encounter UFM 4390, which states that the requested LOADSET does not exist in the SLT. This can be corrected by either adding a dummy LSEQ that is referenced by the case control to the input section referenced or by changing the case control so that the LOADSET does not apply to that superelement.

The following file shows a solution for the same problem using PART superelements using LOADSET-LSEQ.

```
ID seg12-a1, sample
  $ sample to demonstrate loadings on superelements in dynamics
  $
SOL 112
TIME 10000
CEND
  TITLE = SEG11-A1 - CANTILEVER BEAM DIVIDED INTO TWO SUPERELEMENTS
  subtitle = fixed boundary cms - three modes per superelement
  label = transient response with load in superelement 2
  DISPLACEMENT(PLOT) = ALL
  SPC = 1
$
LOADSET = 999
DLOAD = 101
TSTEP = 102
$
SUBCASE 1
```
This file is similar to seg12_a1.dat, except that PARTs are used. The case control is identical to seg12_a1.dat. The only differences occur in the Bulk Data Sections. In the Main Bulk Data Section and in the section for PART 2, there is an LSEQ entry (a copy of the one used in PART 1), which has no associated static loading entries. This is to avoid UFM 4390. In the section for PART 1, we have the LSEQ and the associated loading entry for grid point 11. Remember that PARTs need to be self-contained, so if OLOAD (applied loading) output is desired for PART 1, you must also copy the TLOAD2 and TSTEP entries into its data section. Although these entries are not used as part of the solution, the data recovery operation uses them to generate a time history of the applied loading (if requested).


Siemens Industry Software

Headquarters
Granite Park One
5800 Granite Parkway
Suite 600
Plano, TX 75024
USA
+1 972 987 3000

Americas
Granite Park One
5800 Granite Parkway
Suite 600
Plano, TX 75024
USA
+1 314 264 8499

Europe
Stephenson House
Sir William Siemens Square
Frimley, Camberley
Surrey, GU16 8QD
+44 (0) 1276 413200

Asia-Pacific
Suites 4301-4302, 43/F
AIA Kowloon Tower, Landmark East
100 How Ming Street
Kwun Tong, Kowloon
Hong Kong
+852 2230 3308

About Siemens PLM Software

Siemens PLM Software, a business unit of the Siemens Industry Automation Division, is a leading global provider of product lifecycle management (PLM) software and services with 7 million licensed seats and 71,000 customers worldwide. Headquartered in Plano, Texas, Siemens PLM Software works collaboratively with companies to deliver open solutions that help them turn more ideas into successful products. For more information on Siemens PLM Software products and services, visit www.siemens.com/plm.