ATA Project No. 62038
Customer PO No. 746641/1
October 15, 2013

Final Report for

Customization of Femap FEA Preprocessor for Shipbuilder Needs Including User's Guide

Prepared for

Huntington Ingalls Shipbuilding
PO Box 149
Pascagoula, Mississippi 39568-0149

Prepared by

ATA Engineering, Inc.
13921 Park Center Rd, Suite 340
Herndon, Virginia 20171

Prepared by:

[illegible]
Scanned signature, original on file ATA ENGINEERING INC

Victoria Harris
Project Engineer

Reviewed by:

Christopher Kubik
Scanned signature, original on file ATA ENGINEERING

Christopher Kubik
Director, Eastern Region Operations

CONTACTS

<u>Huntington Ingalls Shipbuilding</u>:	
Jamie Breakfield	228.935.7067
<u>ATA Engineering</u>:	
Victoria Harris	703.988.1951
Chris Kubik	703.796.9181

ABSTRACT

This final report documents the Shipbuilder Toolbox created by ATA Engineering, Inc. (ATA). Huntington Ingalls Industries (HII) led a team that included ATA and Bath Iron Works (BIW), and this team secured a panel project from the National Shipbuilding Research Program (NSRP). Based on requirements defined by HII, BIW, and Gibbs & Cox (G&C), ATA created a Femap toolbox that would improve the process of creating an analysis model from CAD geometry. The tools aim either to automate manual tasks within Femap or provide a shortcut for tedious procedures. Since the most time-consuming part of creating an analysis model is often the creation of meshable geometry, the toolbox contains several programs for creating or editing mid-surface models. The toolbox also includes new methods for checking the quality of a model as well as tools to aid with mesh grouping. The toolbox is available to all members of NSRP at no cost.

TABLE OF CONTENTS

LIST OF FIGURES

1. INTRODUCTION

ATA Engineering has created a “Shipbuilder’s Toolbox” for use with Femap, in collaboration with HII and BIW. The purpose of the toolbox is to reduce the cost for shipbuilders to perform contractually required analyses by improving the methods used to create the analysis models. The project was funded by NSRP as a panel project.

Analysis of a ship can take tens of thousands of labor hours; there are many models that must be created, and there are many steps in the creation of each model. This project aims to reduce the hours required to create each model, either by automating tedious tasks or by creating shortcuts for difficult procedures. Since creating “meshable” geometry is often the most time-consuming part of the process, many of the tools focus on the creation and editing of mid-surfaces. The toolbox also provides programs that check model quality in new ways, as well as a few tools that aid in the actual meshing process, with new grouping methods and beam orientation tools. There are also programs for parsing log files and creating load combinations.

The toolbox uses the Femap Application Program Interface (API); this is a programming tool that can be used to automate almost any Femap manual process. The toolbox was written in Visual Basic and is accessed from within Femap (see Section 2.3 for details on installation). The tools were all written in conjunction with Femap 10.2.1; different versions of Femap may cause small differences in the behavior of the tools.

ATA Engineering will provide the initial release version of the toolbox to any member of NSRP free of charge.

2. ANALYSIS OF A SHIP MODEL USING FEMAP

Many of NSRP’s members use Femap to perform the structural analyses required of them. Often, the CAD geometry is created in a different program and imported to Femap, which complicates efforts to prepare the geometry for meshing. The following sections describe some of the frustrations of the current tools and explain how customization is used to advance their capabilities.

2.1. Current Analysis Methodology

Structural analysis using finite element models requires the analyst to go from geometry to mesh, but there are multiple ways to approach this task. ATA and HII worked together to describe a typical workflow for an HII analyst; this workflow is shown in Figure 2-1.

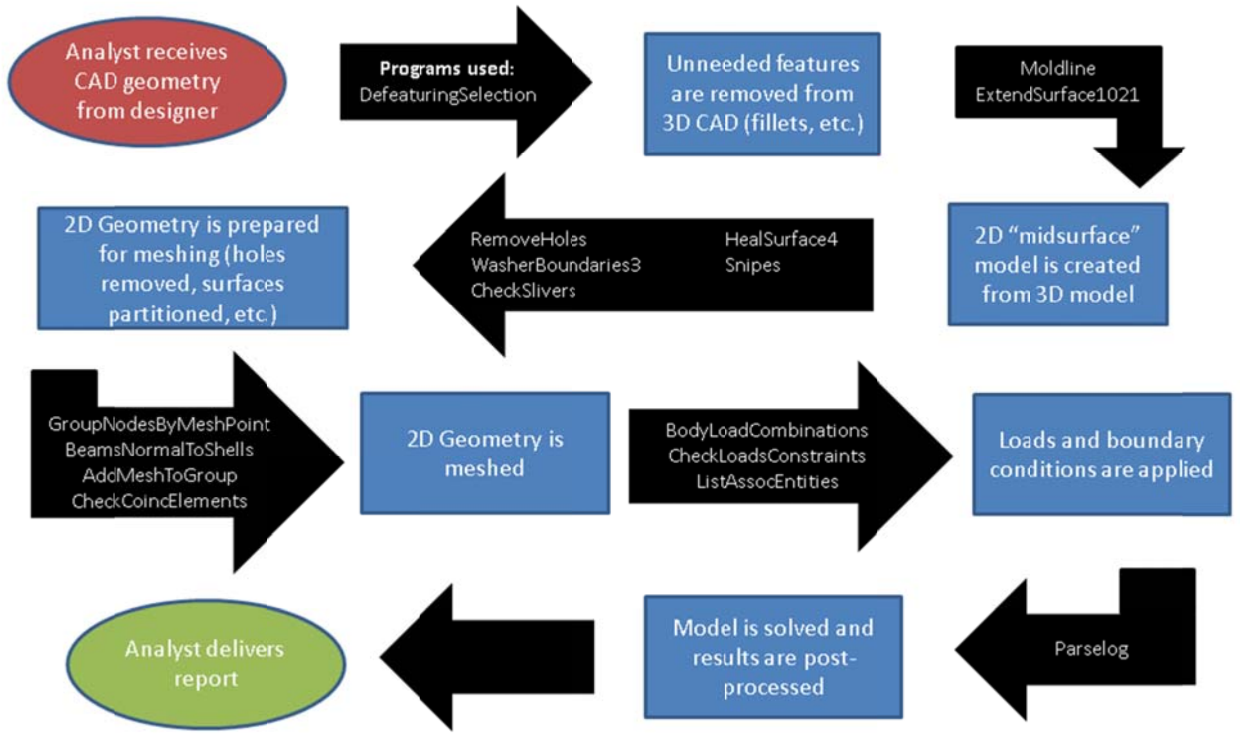

Analyst receives CAD geometry from designer
Programs used: DefeaturingSelection
Unneeded features are removed from 3D CAD (fillets, etc.)
Moldline ExtendSurface1021
2D “midsurface” model is created from 3D model
RemoveHoles WasherBoundaries3 CheckSlivers HealSurface4 Snipes
2D Geometry is prepared for meshing (holes removed, surfaces partitioned, etc.)
GroupNodesByMeshPoint BeamsNormalToShells AddMeshToGroup CheckCoincElements
2D Geometry is meshed
BodyLoadCombinations CheckLoadsConstraints ListAssocEntities
Loads and boundary conditions are applied
Parselog
Model is solved and results are post-processed
Analyst delivers report

Figure 2-1. Typical workflow in Femap for HII analyst. The blue boxes show steps taken by the analyst; the black arrows list the toolbox programs that are useful for each step.

Typically, the analyst receives CAD geometry from the designer in a neutral file format, such as .step or .iges. This geometry usually consists of detailed 3-D entities with features such as fillets and fastener holes. Figure 2-2 shows an example of a generic ship section provided by HII; the deck panels and stiffeners are all represented with 3-D entities, and many of the stiffeners have features like fillets and snipes.

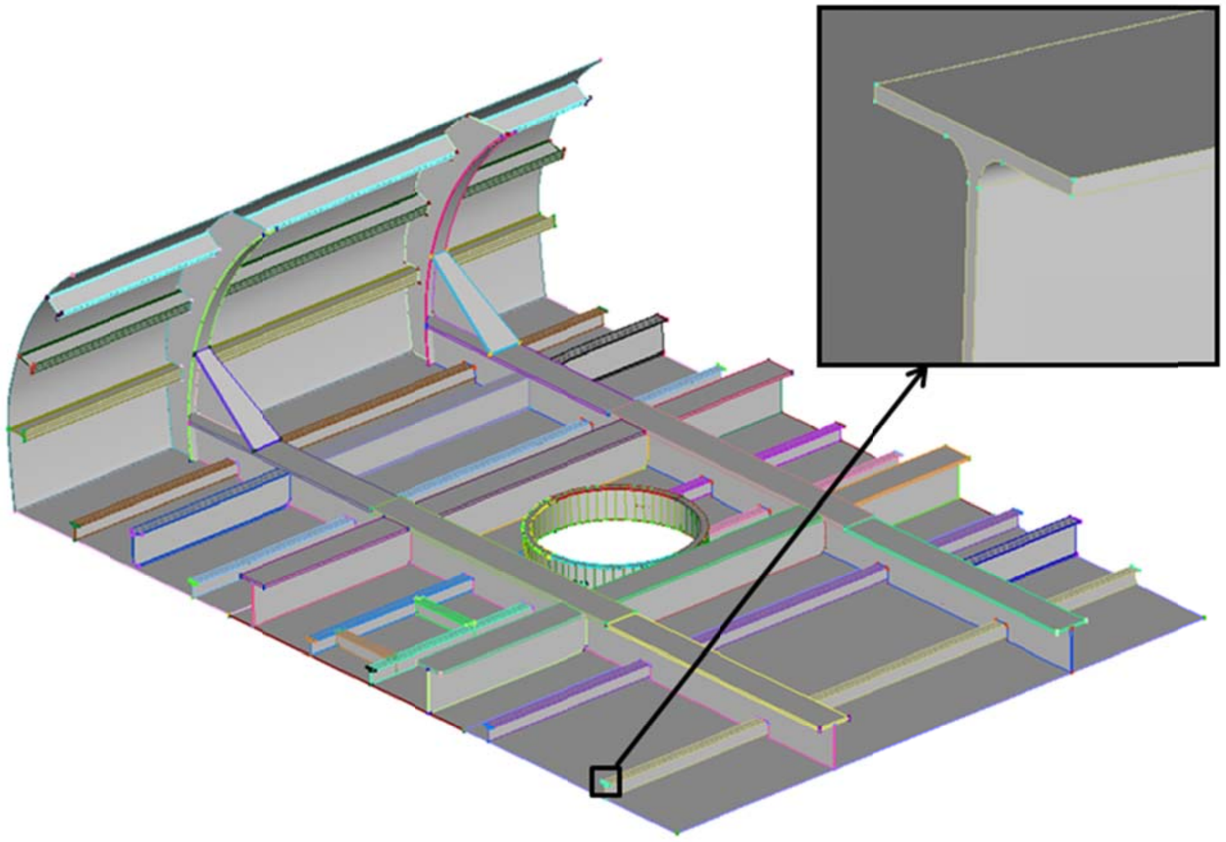


Figure 2-2. Typical CAD geometry received from designer; inset shows fillets on a stiffener.

Large, thin-walled structures like ships are typically meshed using 2-D shell elements; this method reduces the number of nodes in a model, which improves solve time, but requires a lot of geometry preparation. Geometry preparation is often the most time-consuming step, and most of the tools described in later sections focus on this phase of the meshing process.

To produce a mesh, the 3-D entities must first be represented with 2-D surfaces that can be meshed with 2-D elements. Femap has a mid-surfacing tool for creating 2-D surfaces from solids, but it works better when design-level details like fillets are removed ahead of time. Femap has some tools for feature suppression and removal, but it can be tedious, even on a model as small as the one shown in Figure 2-2, to select every feature that needs to be removed.

Once the geometry has been defeatured, the mid-surfacing tool can be used to create 2-D surfaces. This tool is useful for generating a large number of surfaces quickly, but those surfaces typically need a lot of time-consuming cleanup. For instance, solid panels of different thicknesses will result in mid-surfaces that are not coplanar, as shown in Figure 2-3. In other places, surfaces will not intersect where they should and will need to be extended (also shown in Figure 2-3).

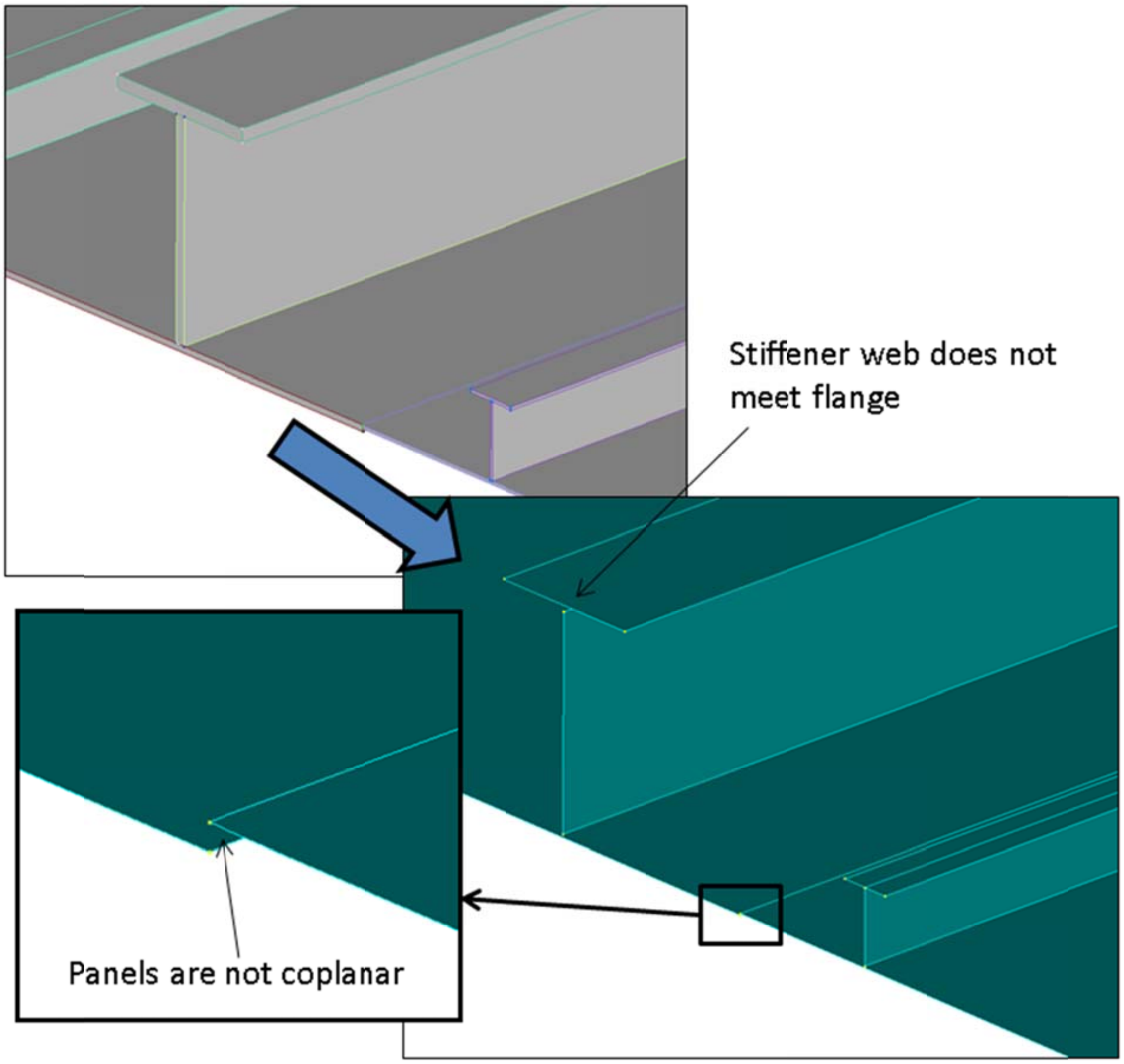


Figure 2-3. The mid-surface tool produces surfaces quickly, but the model still needs work.

Once the surfaces are positioned and connected correctly, they can be prepared further for meshing. For example, the user may want to either combine or partition surfaces to provide better mesh control, or remove holes. This is also when the user can check element quality.

Once the geometry preparation is complete, the surfaces can be meshed; the mesh is typically organized into groups at this point. Next, the boundary conditions are set up and the solution is prepared, and finally the model is solved and the results analyzed.

2.2. <u>Customization of Femap</u>

For customization, Femap provides the Application Program Interface (API). The API can be used to automate almost any Femap process, and multiple programming languages can be used with the interface, including Visual Basic and C++. The resulting programs can interact with other software such as Excel and Matlab. The custom programs can interact with Femap from within a session, or Femap can be embedded within the program.

For this toolbox, all programs were written in Visual Basic and operate from within Femap. The following section describes how to install and use the tools.

2.3. <u>Installation of Shipbuilder Toolbox</u>

The Shipbuilder toolbox is provided as a folder containing executable files. The tools can be added to Femap in two ways:

First, tools can be added to Femap one at a time. In Femap, go to Custom Tools → Add Tools… and then navigate to the location of the executable for the tool of interest and select that .exe file. Femap will copy the .exe file to its Tools Directory (in the Femap installation) and add the name of the program to the Custom Tools list. The program can then be accessed from this list.

The entire toolbox can also be added to Femap. First, go to Custom Tools → Tools Directory… and note the Tools path (e.g., C:/Siemens/FEMAPv1031/api/). Then, using Windows Explorer (or similar), copy the entire Shipbuilder folder provided by ATA into the Tools Directory. The name of the Shipbuilder folder should show up in the Custom Tools list with a submenu containing all of the programs, as shown in Figure 2-4.

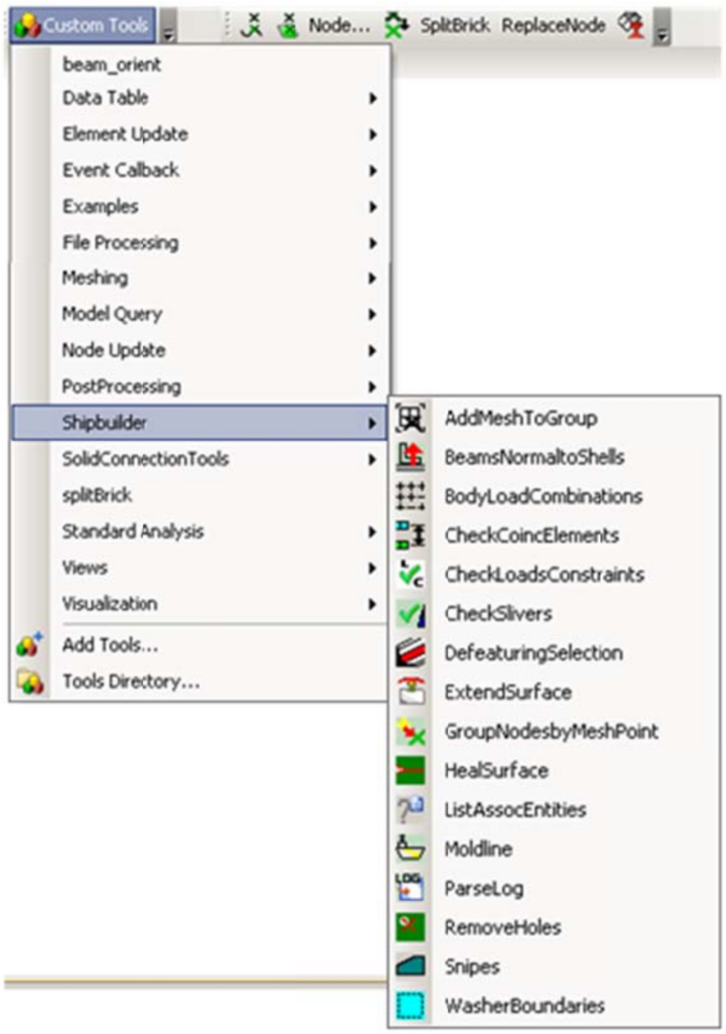


Figure 2-4. When added to the Tools Directory, the Shipbuilder toolbox shows up as a submenu in the Custom Tools list.

3. MID-SURFACING TOOLS

The most time-consuming step in the analysis process is often the creation of "meshable" geometry. The most common way to model large, thin-walled structures like ships is with 2-D elements. To be able to create these 2-D meshes, the analyst must convert the 3-D geometry provided by the designer into 2-D surfaces; typically this conversion is achieved by using mid-surfaces.

Femap has a set of built-in tools to create surfaces at the mid-planes of solids. These tools are useful for creating the basic surfaces, but analysts must make a lot of manual adjustments to make the surfaces useful. The following sections describe programs designed to automate some of the more tedious aspects of these manual adjustments.

3.1. DefeaturingSelection

This program is intended to assist in simplifying solid geometry before attempting to mid-surface it, by speeding up the process of selecting surfaces that meet a specific criterion. The user selects solids and the program searches for fillets or other small surfaces on the solids. These surfaces can then be passed to Femap's feature removal tool in the meshing toolbox. However, the selected surfaces may or may not be removable with the feature removal tool; it is up to the user to inspect the selected surfaces before attempting to remove them.

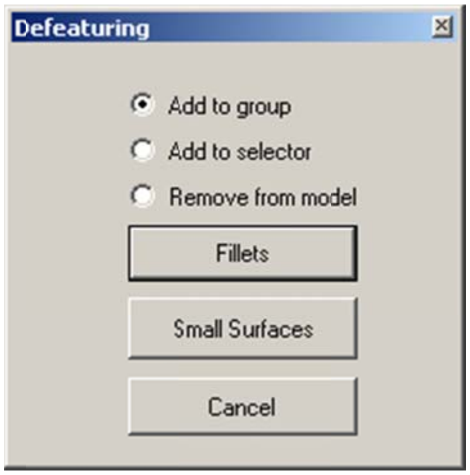


Figure 3-1. DefeaturingSelection dialog.

Surface Collection Options:

The DefeaturingSelection tool provides several options, shown in Figure 3-1:

Add to group: Creates a new group and puts the surfaces in it. The group is named either "Fillets" or "Small Surfaces" depending on the option selected.

Add to selector: Adds the surfaces to the selector.

Remove from model: Passes the selected surfaces directly to the feature removal tool in the meshing toolbox. The program will attempt to remove surfaces in order from largest area to smallest area.

Surface Type Options:

Fillets: Looks for surfaces with a radius smaller than the tolerance set by the user. This tool will skip any curved surfaces belonging to holes.

Small surfaces: Looks for surfaces with a maximum edge length smaller than the tolerance set by the user.

3.2. **Moldline**

This program improves the mid-surfacing process by allowing the user to designate certain surfaces as mold-lines when mid-surfacing a model. This results in a model with some surfaces located at the mold-line surface of a solid and some surfaces located at the mid-plane of a solid. See Figure 3-2 through Figure 3-5 for an example.

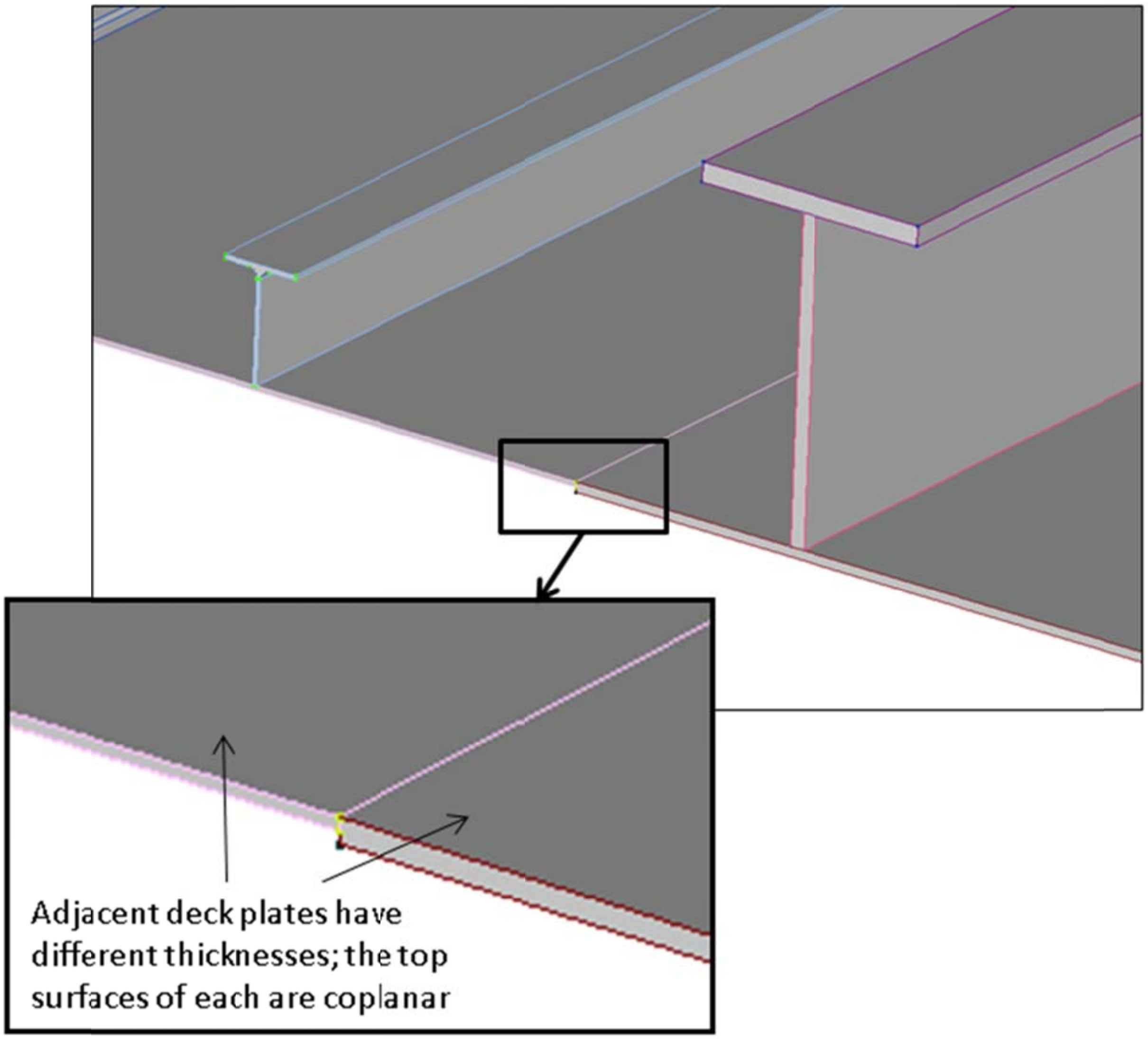

Adjacent deck plates have different thicknesses; the top surfaces of each are coplanar

Figure 3-2. Solid geometry.

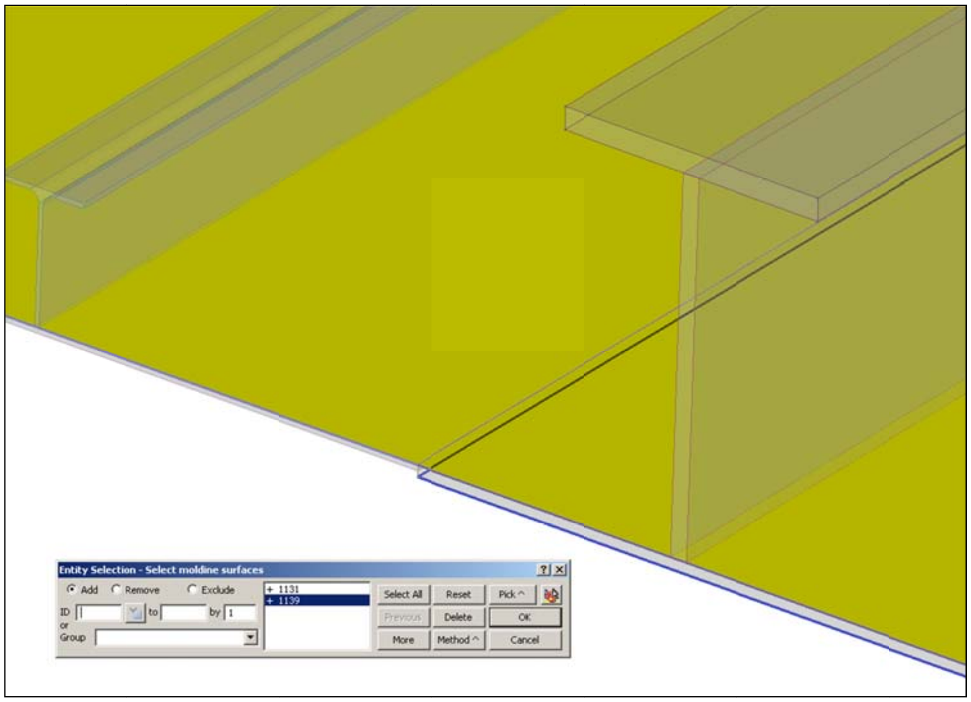


Figure 3-3. Mold-line surfaces are selected (highlighted in yellow).

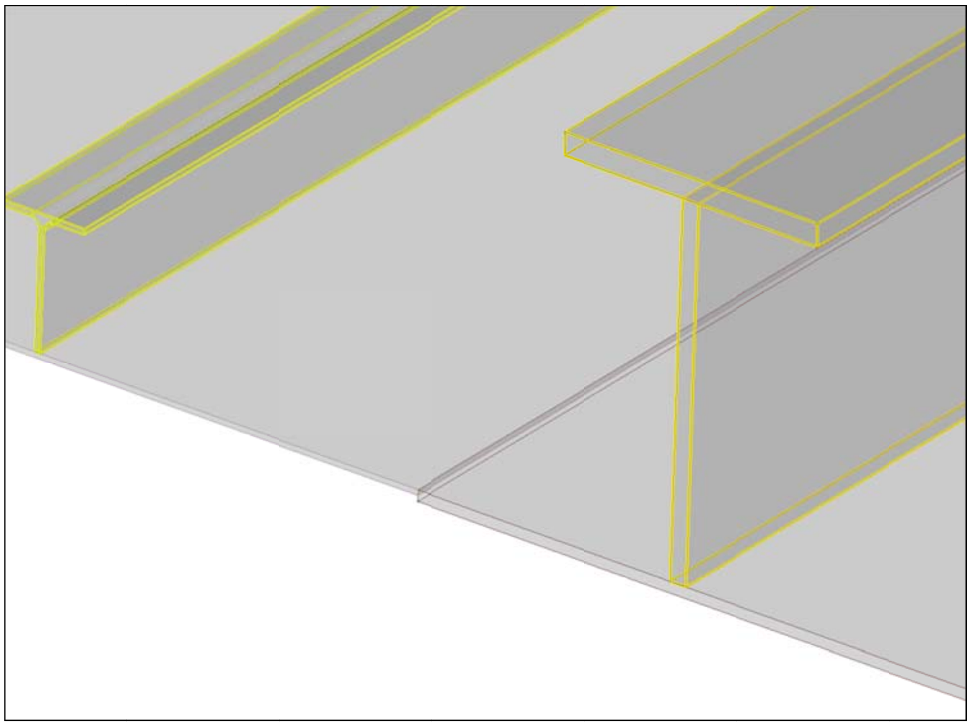


Figure 3-4. Solids are selected for mid-surfacing (highlighted in yellow).

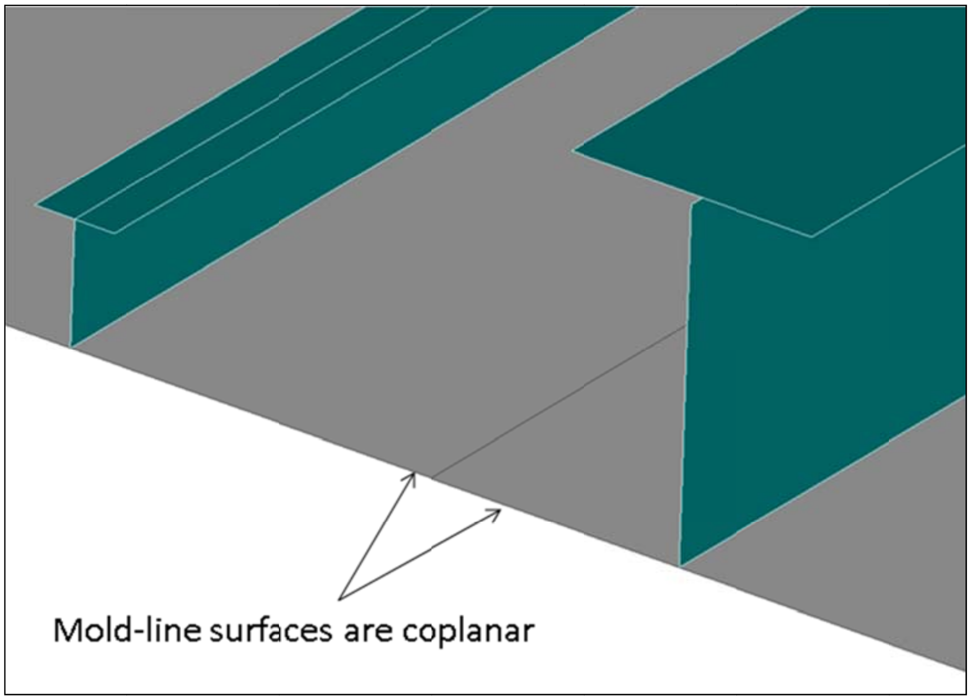


Figure 3-5. Mold-lines and mid-surfaces have been created.

The user must specify a thickness tolerance for mid-surfaces. This part of the program uses the built-in Femap mid-surfacing tool, and the tolerance is used to search for appropriate face pairs. The Moldline program calculates a thickness for each mid-surface and mold-line surface and assigns each an appropriate 2-D shell property. All of the newly created surfaces are placed into a new group called "Moldlines and midsurfaces."

Note: If the program calculates a thickness for a mold-line surface that is greater than the mid-surface thickness tolerance, it will create the mold-line surface but will not assign it a thickness.

3.3. RemoveHoles

RemoveHoles removes holes from selected surfaces according to criteria set by the user. This function differs from the built-in Femap function for removing holes in that the Femap function only looks for holes that are defined by specific curve types (arcs or circles), while this program removes any closed loop in a given surface, circular or otherwise.

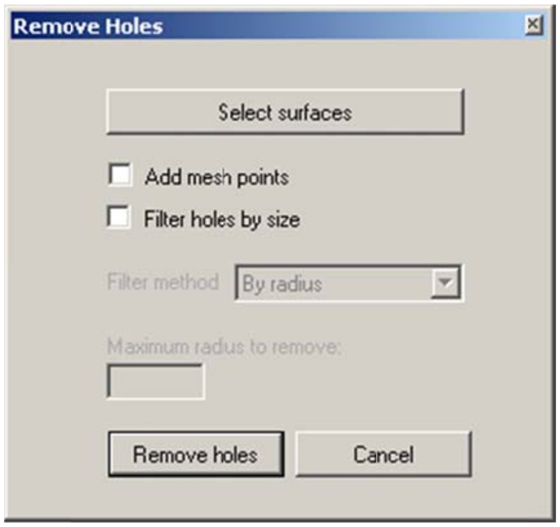


Figure 3-6. RemoveHoles dialog.

RemoveHoles Options:

Figure 3-6 shows the RemoveHoles options:

Select surfaces: User selects surfaces from which to remove holes.
Add mesh points: Adds a mesh point at the center of each removed hole.
Filter holes by size: Remove only the holes below a certain threshold.

By radius: Set a radius tolerance. Only holes with a radius smaller than the tolerance will be removed. For noncircular holes, the radius is calculated to be the length of the perimeter divided by 2π.

By circumference: Only holes with a circumference smaller than the tolerance will be removed.

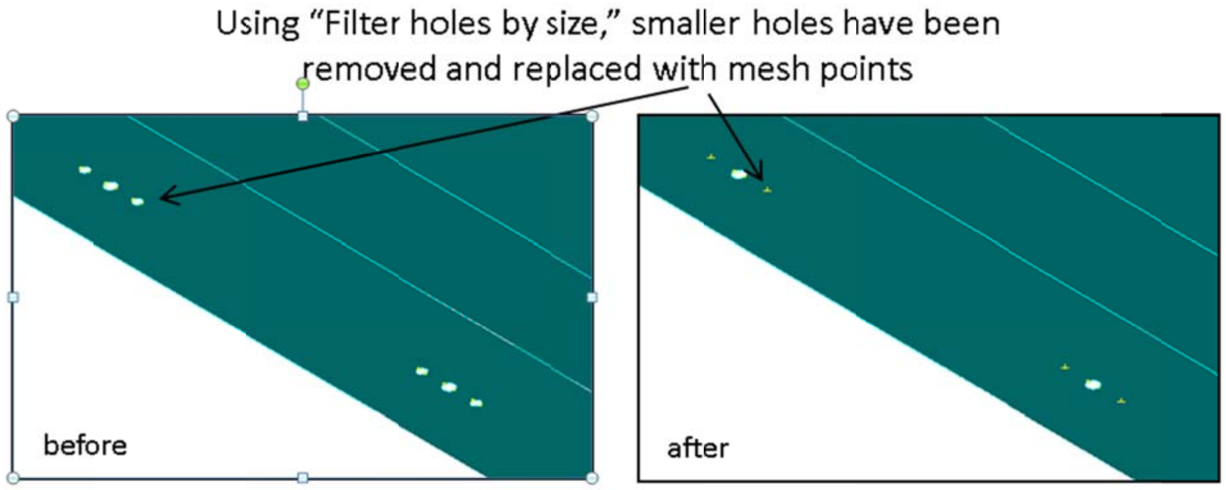

Using "Filter holes by size," smaller holes have been removed and replaced with mesh points
before
after

Figure 3-7. Use the size filter to selectively remove holes in surfaces.

3.4. <u>HealSurface</u>

This program combines multiple surfaces into a single surface. If the surfaces are on different solids, the program will ask whether to combine those solids before it can merge the surfaces (if the user says no, the surfaces will not be combined). Internal loops (e.g., holes) will be carried over. Figure 3-8 shows a set of surfaces in the top picture (marked by asterisks) that are combined into the single, blue-green surface shown in the bottom picture.

Note: Any data attached to the original surfaces (properties, loads, etc.) is not copied to the new surface. The program does not check to see if selected surfaces are tangent at the shared edge, so nontangent surfaces may produce unusual results. In Femap 10.2.1, there is no option to set a tolerance for merging solids, so if the selected surfaces are not close enough together according to the Femap default (1e-6), the program will not be able to merge them.

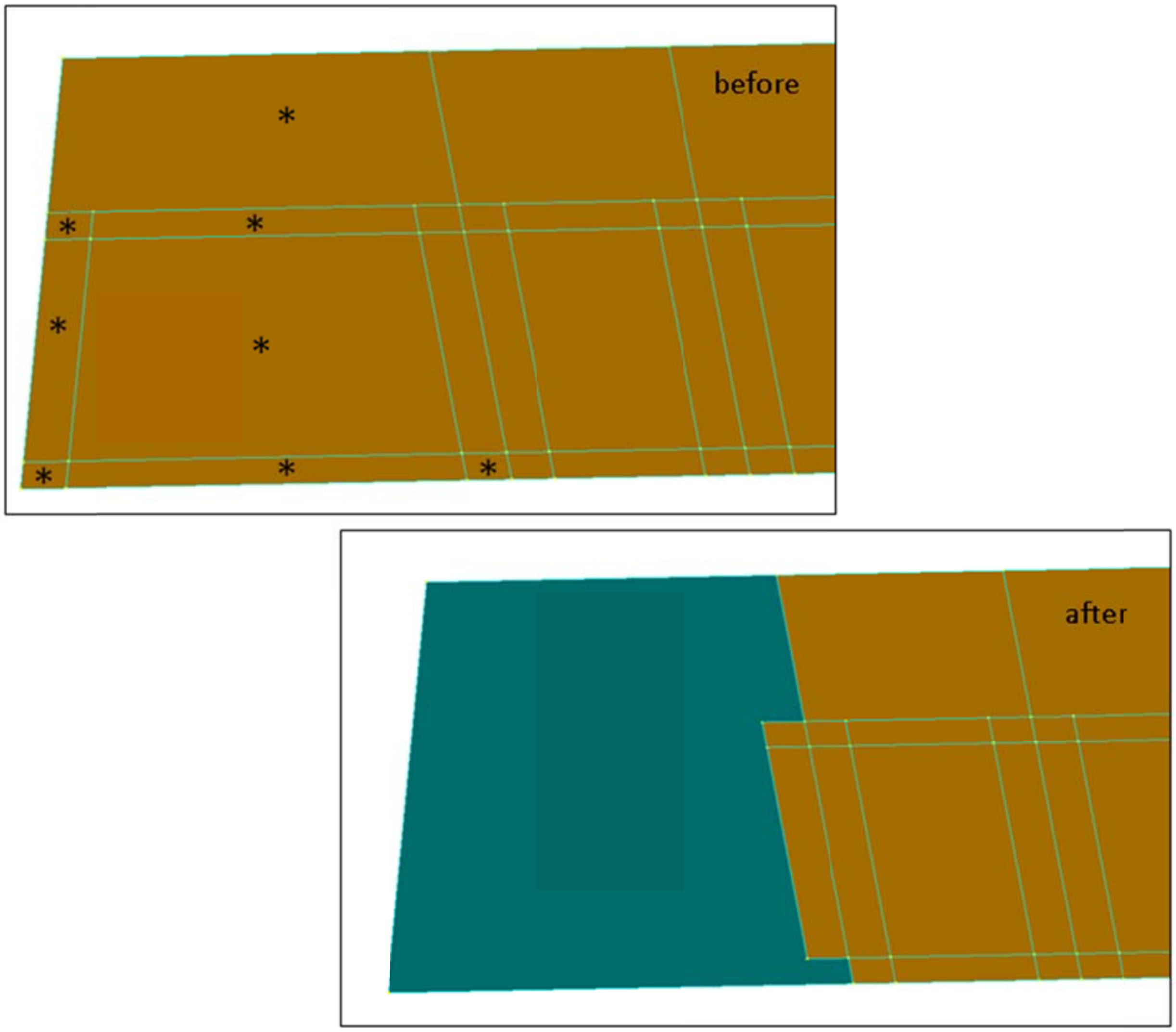


Figure 3-8. Surfaces with asterisks in top picture are combined using HealSurface to produce bottom picture.

3.5. WasherBoundaries

This program partitions a selected surface by offsetting that surface's edges into the surface itself, essentially creating a border as shown in Figure 3-9.

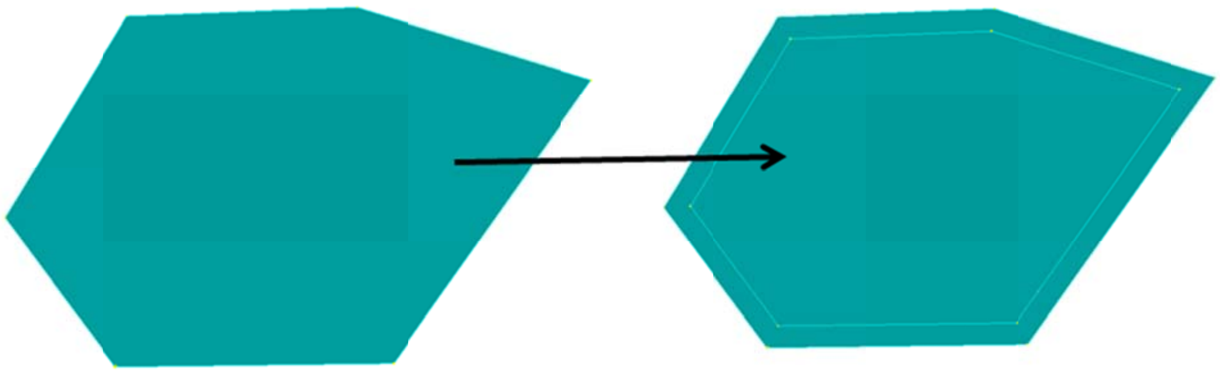


Figure 3-9. WasherBoundaries used to create a border on a surface.

After the user has selected surfaces, WasherBoundaries asks the following questions:

Imprint internal surface boundaries (holes, etc.)?: If yes, the program will attempt to create a border around any internal loops in the surface, like a hole.

Create corner imprints as separate surfaces?: If yes, the program will extend the split lines used to create the borders so that the corners are broken off from the rest of the border as shown in Figure 3-10.

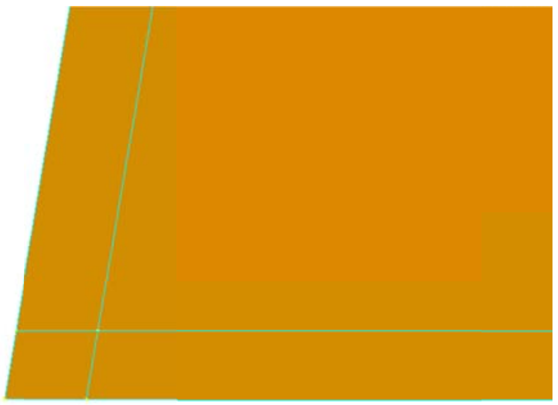


Figure 3-10. Corner imprint.

Offset Distance: User defines the width of the border.

Note: In Femap 10.2.1, sometimes the borders are created at a distance equal to the user-specified offset divided by √2. If the user encounters this problem, they must undo the first attempt at a washer boundary and redo it with the offset distance multiplied by a factor of √2.

3.6. ExtendSurface

ExtendSurface extends a selected surface up to a second surface. This program expands upon the surface-extend capability already present in Femap by allowing the user to select a target surface that is smaller than the surface to be extended. If the target surface is planar, it is used to define a plane to which the surface is extended. If the target is not planar, it is extrapolated to provide a curved target for the extending surface to reach. An example is shown in Figure 3-11: the extending surface never intersects the target surface, but a planar target is inferred from the target surface. After the surface is extended, HealSurface can be used to merge the new surface with the original.

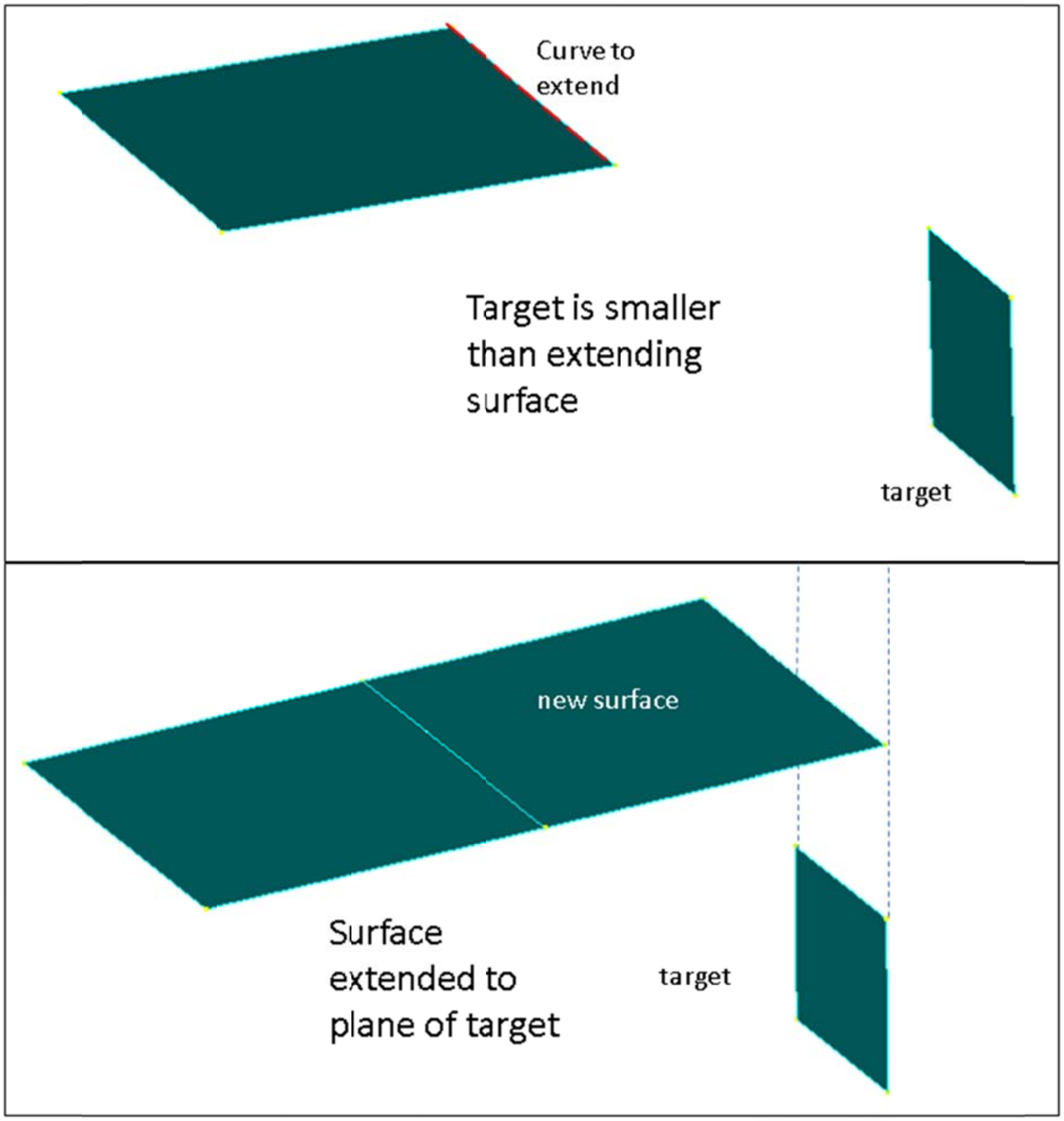

Curve to extend
Target is smaller than extending surface
target
new surface
Surface extended to plane of target
target

Figure 3-11. ExtendSurface extends surfaces using small target surfaces to define planes.

Note: This program does not work for all cases—for instance, surfaces with a relatively small radius of curvature make poor targets. In addition, in some cases the tool will produce two surfaces: the surface extension intended and an extra "overhanging" piece. This is because the program is designed to err on the side of caution when deleting surfaces. The user can delete the overhanging surface after using the program.

3.7. Snipes

The Snipes tool cuts a "snipe" from the corner of a rectangular surface. The tool asks the user to identify the corner in question by selecting a point. The program then asks the user for the primary edge, which is used to define the other two parameters—the snipe depth, which is the height of the remaining surface after the snipe is cut, and the snipe angle, which is the angle of the cut. These parameters are all shown in Figure 3-12; the diagram in this figure is also displayed while the Snipes tool is in use.

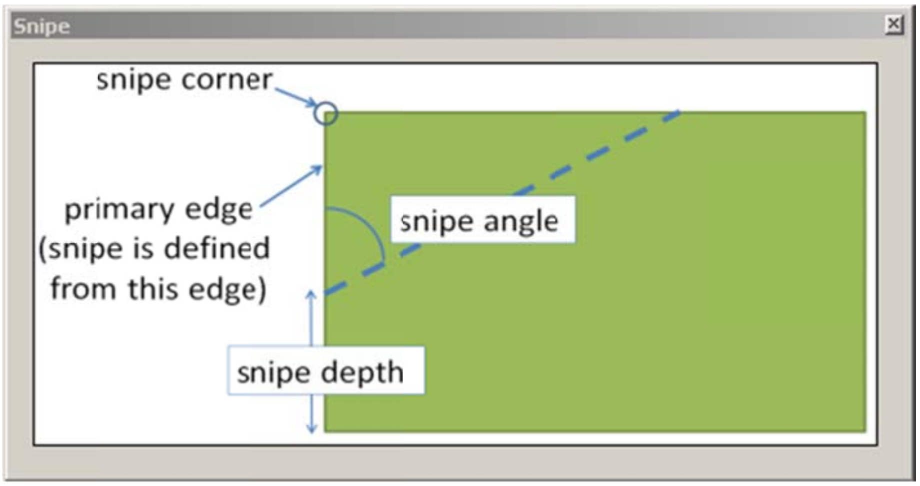


Figure 3-12. Snipe parameters.

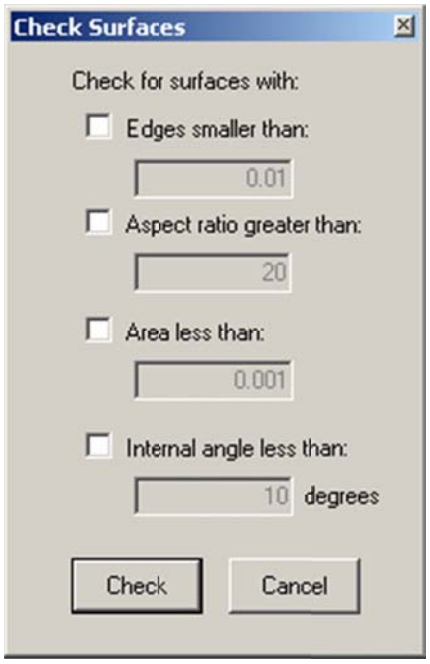
Note: This program will not work if there is more than one surface attached to the selected snipe corner. If the angle at the snipe corner is very acute or very obtuse, the program may not produce the desired result.

4. <u>MODEL CHECKS</u>

The programs in this section can be used at various stages of the analysis process to check a model for issues. CheckSlivers looks for surfaces that have undesirable features so that they can be fixed or removed before meshing. CheckLoadsConstraints looks for geometry that has boundary conditions assigned to it but no mesh so that the user can find parts of a model where multiple geometric entities may be overlapping. ListAssocEntities provides more information about associations between entities, and CheckCoincElements provides a more thorough inspection of overlapping elements.

4.1. <u>CheckSlivers</u>

The CheckSlivers program identifies surfaces in a model with features that may cause problems during meshing. The user selects which surfaces to include in the search and also which criteria to apply. Surfaces that meet the search criteria are added to a new group labeled "Sliver surfaces." Search criteria can be combined in different ways; the best combination of criteria depends on the features of the model being checked.


Check Surfaces

Check for surfaces with:

☐ Edges smaller than: 0.01

☐ Aspect ratio greater than: 20

☐ Area less than: 0.001

☐ Internal angle less than: 10 degrees

Figure 4-1. CheckSlivers dialog.

Figure 4-1 shows the criteria:

Edges smaller than: The user sets an edge length, and the program identifies any surfaces with an edge smaller than that length.

Aspect ratio greater than: This option compares the length of every edge on a given surface to the length of every other edge on that surface. If the ratio between the two is smaller than that set by the user, the surface is flagged. This option is intended to identify long, thin sliver surfaces; if a surface has small details on it (like filleted corners), then it may get flagged unnecessarily.

Area less than: The user sets an area threshold, and surfaces with an area smaller than the threshold are flagged.

Internal angle less than: This option checks the angle between intersecting edges at all points on a surface. If any angles are smaller than the user-selected criterion, then the surface is flagged. This option is useful for finding long triangular slivers.

4.2. <u>CheckLoadsConstraints</u>

The CheckLoadsConstraints program checks the model for any loads or constraints defined on geometry that has no associated mesh. The motivation for this program is a situation in which two pieces of identical geometry (e.g., two curves) overlie each other so that it is impossible to tell just by looking that there are two curves instead of just one. When this situation arises in a model, sometimes the loads or constraints are defined on one instance of the curve while the mesh is applied to the other, and the user does not discover this problem until they attempt to solve the model, in which case either the solve fails or the results are calculated with the wrong boundary conditions.

This program allows the user to discover the problem before attempting to solve the model. The user selects load or constraint sets, and the program lists any geometry referenced by the loads/constraints that does not have an associated mesh. The program also adds the geometry in question to a group, labeled "Geometry/Constraints with no mesh." The user can then edit the defined boundary conditions so that they reference geometry with associated mesh.

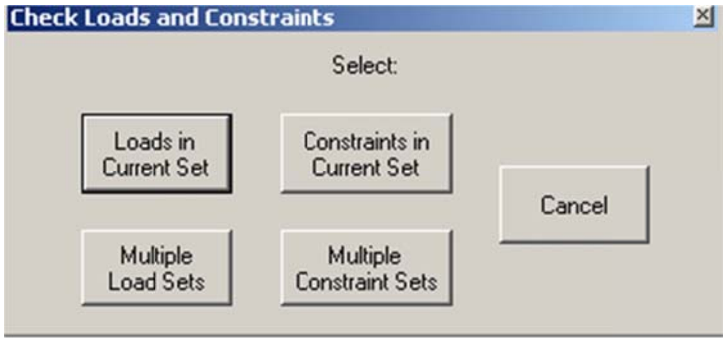


Figure 4-2. CheckLoadsConstraints dialog.

Figure 4-2 shows the dialog box for this program. The user can choose between selecting individual loads and constraints in the set currently active in the model or selecting all the loads/constraints from multiple sets.

4.3. <u>ListAssocEntities</u>

This program is intended to aid the user in figuring out connections between different Femap entities that are not immediately accessible using the List tools. (The term "Femap entity" can refer to geometry, mesh, or other "objects" in Femap such as curves, elements, loads, etc.). For instance, this tool can be used to list any geometric entity that has a particular material assigned to it. This is useful when a user tries and fails to delete a material from the model, as the user can quickly determine what entities reference the material and then either delete those entities or remove the references so that the material can be deleted.

To limit the amount of data written to the message window, the types of associated entities listed are limited. For geometry entities, all other associated geometric entities are listed. For all geometry entities except solids, the associated loads and constraints are listed too. For materials and properties, all geometric entities are listed, and for materials, associated properties are listed as well. For all selected entities, if there is no associated geometry, that fact is listed in the message window.

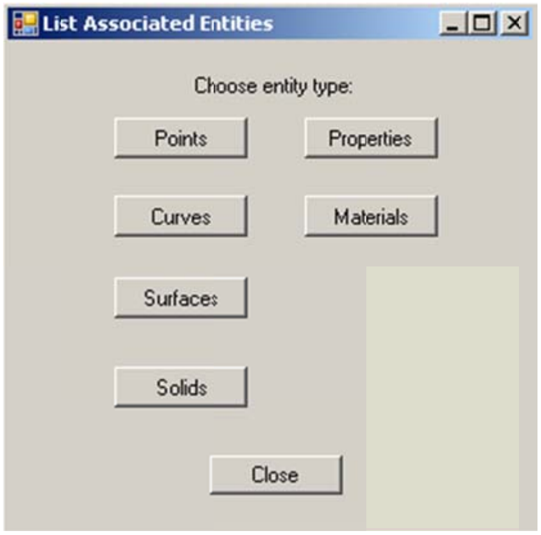


Figure 4-3. ListAssocEntities dialog.

Figure 4-3 shows the dialog box for ListAssocEntities, which asks the user to select an entity type. The button labels refer to the entity the user may wish to select; for instance, clicking on Points will prompt the user to select one or more points, and then the program will list the entities associated with the selected points.

4.4. <u>CheckCoincElements</u>

CheckCoincElements is an expansion of the existing coincident-element check in Femap. Femap's coincident-element check looks for any two elements that share all of their nodes. CheckCoincElements looks for elements that occupy the same location without necessarily sharing any nodes; the program will identify any element whose nodes are all coincident with another element's nodes. The second element must be the same type as the first element, and both elements are added to new groups.

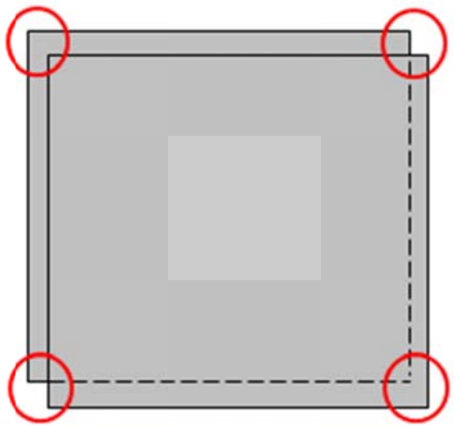


Figure 4-4. CheckCoincElements looks for any element whose nodes are all coincident with another element's nodes.

The user selects elements to check and a tolerance by which to define coincidence. The program takes all the nodes on those elements and checks for any coincident nodes. If any coincident nodes are found using the tolerance, the elements attached to those nodes are added to groups (for each pair of coincident nodes, the elements attached to one node go in the first group while the elements attached to the other node go in the second group). Then each element in the first group is compared to each element in the second group, and if every node on the first element is close enough to a node on the second element, the elements are added to the final coincident elements groups (labeled “Coincident elements 1” and “Coincident elements 2”). For instance, Figure 4-4 shows two overlapping elements—each corner of the first element is very close to a corner of the second element. These elements would be considered coincident.

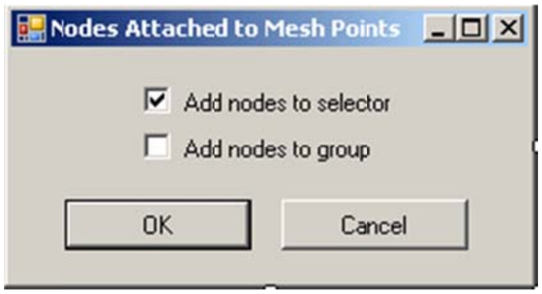
Note: In certain situations, this program will identify elements that are intentionally coincident (perhaps two layers of shell elements represent surfaces of two different but adjacent objects). For this reason, it is recommended that the user carefully examine each group of coincident elements before deciding to delete one or the other.

5. MESHING TOOLS

The following tools are useful during and after the actual meshing process. GroupNodesByMeshPoint and AddMeshToGroup automate grouping processes, while BeamsNormalToShells automates beam orientation.

5.1. GroupNodesByMeshPoint

This program puts any nodes attached to mesh points into the selector tool and/or a group. The user selects surfaces and the program identifies any mesh points on the surfaces and then any nodes on those mesh points. If the group option is selected, the group is named "Nodes attached to mesh points." These options are shown in Figure 5-1.



[✓] Add nodes to selector
[ ] Add nodes to group

Figure 5-1. GroupNodesByMeshPoint dialog.

5.2. AddMeshToGroup

This is a simple program that adds a mesh to the active group. The user selects elements, and AddMeshToGroup adds those elements and all nodes on those elements to the active group. This shortens the usual two-step process of adding elements to a group then adding the nodes on the elements.

5.3. BeamsNormalToShells

BeamsNormalToShells sets the orientation of beam elements using the normal vectors of any shell elements attached to those beams. This has the effect of orienting a beam perpendicular to the "surface" of the 2-D elements to which it is attached. This is similar to the API that orients beams perpendicular to a surface but can be used in situations where the geometry is unavailable.

After the user selects beam elements, the program looks for any shell element that shares a node with a given beam element. Figure 5-2 shows a tube composed of green shell elements with brown beam elements running down the sides; each beam element is attached to several 2-D elements. If there are multiple shell elements attached to the beam element, the normal vectors are averaged. The resulting vector is then assigned as the orientation of the beam element. The final image in Figure 5-2 shows an end-on view of the tube; the I-beams are oriented along an averaged normal vector.

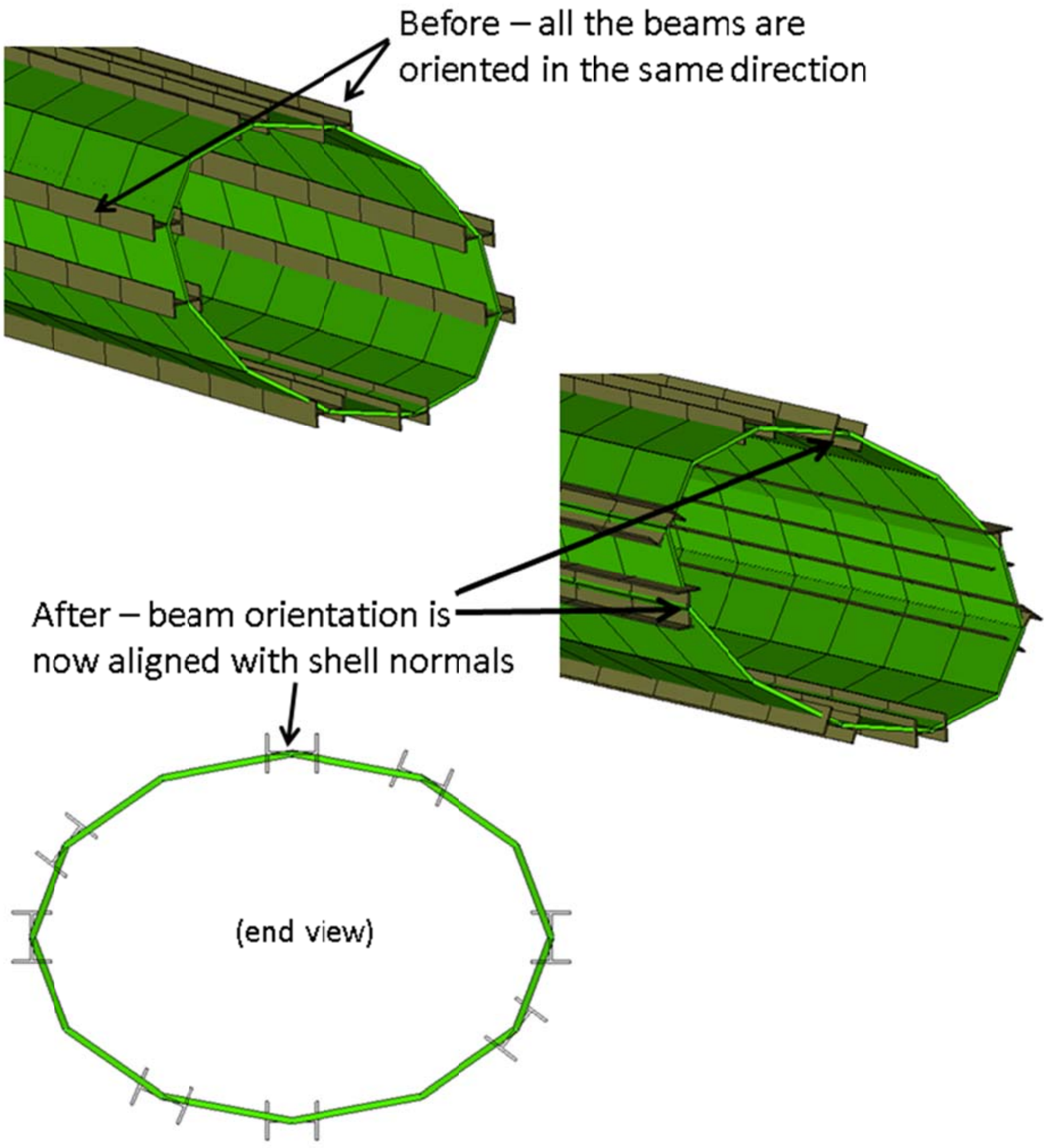

Before – all the beams are oriented in the same direction
After – beam orientation is now aligned with shell normals
(end view)

Figure 5-2. BeamNormalToShells aligns a beam's orientation to the average normal vector of the shell elements attached to the beam element.

Note: This program does not perform any kind of check on the normal vectors of the attached shell elements—it just takes whatever normal vectors exist. For this reason, it is recommended that the user check the normal vectors of the shells before running the program to ensure that they are oriented in the expected manner.

6. MISCELLANEOUS

The programs described below cover other areas of the analysis process; ParseLog creates groups from element warnings in a log file, while BodyLoadCombinations automates the creation of body loads.

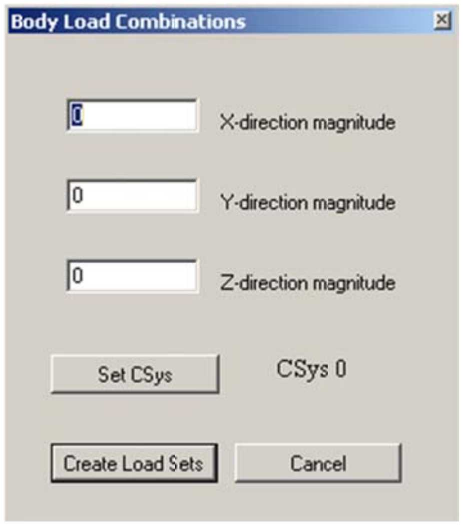
6.1. ParseLog

ParseLog reads in an NEI Nastran .log file and scans it for any element warnings and errors, and then puts the listed elements into a new group. Elements with warnings and elements with errors are written to different groups, labeled “Warning elements from log file” and “Error elements from log file.” If the program finds no element warnings or errors, the user is notified in the message window.

Note: This program only works on NEI Nastran log files; other versions of Nastran produce differently formatted files.

6.2. BodyLoadCombinations

BodyLoadCombinations creates eight sets of body loads containing all possible sign combinations of a given set of magnitudes. With this program, the user supplies a magnitude for a body load in each ordinate direction, as shown in Figure 6-1, and the program creates a load set for each possible sign combination of the three magnitudes. The user can set which coordinate system is used to define the ordinate directions.



0 X-direction magnitude
0 Y-direction magnitude
0 Z-direction magnitude
CSys 0

Figure 6-1. BodyLoadCombinations dialog.

For example, magnitudes of 1,1,1 would produce the following eight load cases:

a) +1 (X-dir), +1 (Y-dir), +1 (Z-dir)
b) +1,+1,−1
c) +1, −1,+1
d) +1, −1, −1
e) −1,+1,+1
f) −1,+1, −1
g) −1, −1,+1
h) −1, −1, −1