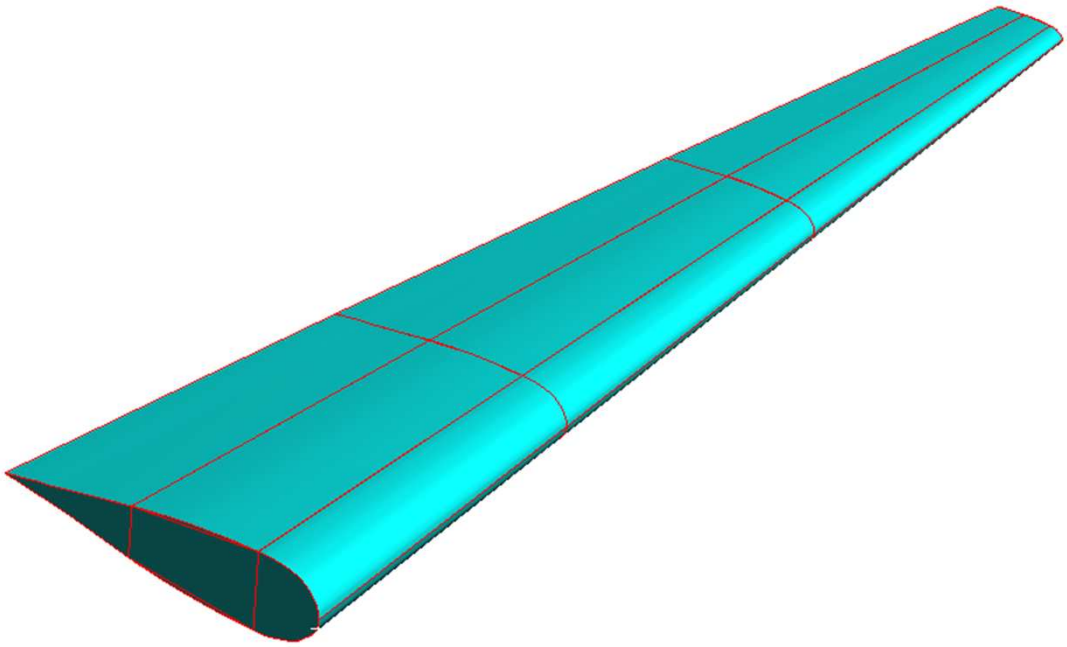
Wing Creation using PCL / PATRAN

DMSM / ISAE

April 2011



Francisco Habib Issa Mattos (ITA/Supaero)
Dr. Joseph Morlier

1.1 Table of Contents

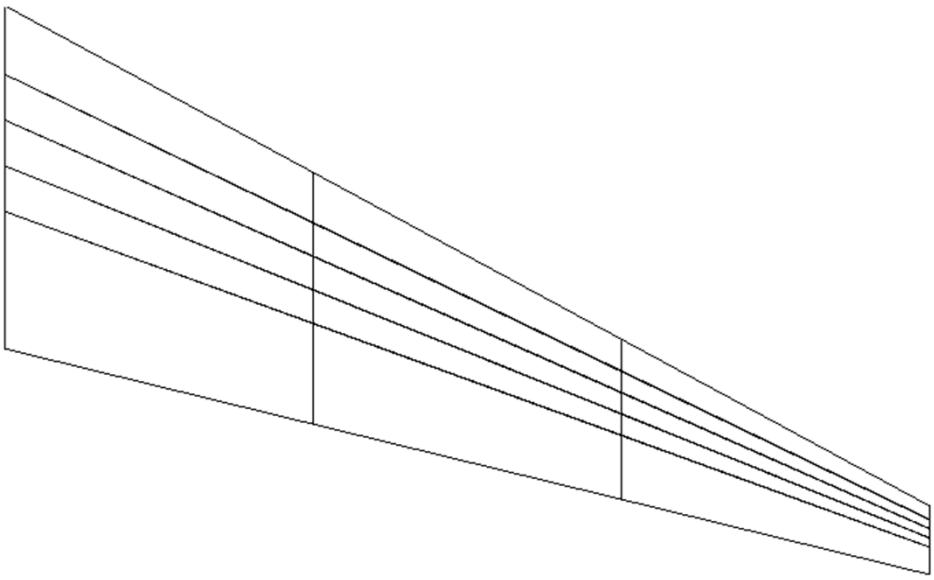
1.1 Problem Description

This tutorial demonstrates how to create, mesh, load and analyze a parametric wing using MSC PATRAN/NASTRAN and PCL code. The first part shows how to create the wing based on parameters such as span, chord sweep, taper ratio, tip torsion, dihedral and a given profile. The second shows how to mesh this wing, and finally the third part shows how to load and analyze the results.

To define a wing we need 1 dimensional + 5 non-dimensional parameters and the cross section, known as the profile. So our inputs are:

- profile.dat
- Wingspan (b)
- Aspect Ration (AR)
- Taper Ratio (λ)
- Dihedral (δ)
- ¼ Chord Sweep (Λ)
- Wing Tip Torsion (θ)

We also have to define the materials and the properties of this materials in the wing. In this tutorial steel, aluminum and titanium are the materials chosen to build the wing. For the properties, shell and bean elements will be used, and for the boundary conditions we will define a fixed wing – cantilever – and analyze it for static loads and normal modes.



We will create a wing with the following parameters, knowing that changing the wing geometry using PCL code does not take more than a minute.

- Wingspan = 30.00 m
- Aspect Ration = 9.0
- Taper Ratio = 0.2
- Dihedral = +6°
- ¼ Chord Sweep = 25°
- Wing Tip Torsion = -3°
- Number of Ribs = 4

The profile used is the one found in Boeing 737 wing root normal cross section. This profile will be the same along the entire wing.

1.2 Geometrical Definitions

In this session we define all the geometrical parameters that will be used during the wing's creation.
Basic parameters:

- Wingspan = Span
- Aspect Ration = AR
- Taper Ratio = TRatio
- Dihedral = Dihedral
- ¼ Chord Sweep = Sweep25
- Wing Tip Torsion = Torsion
- Number of Ribs = nRibs

Derived parameters:

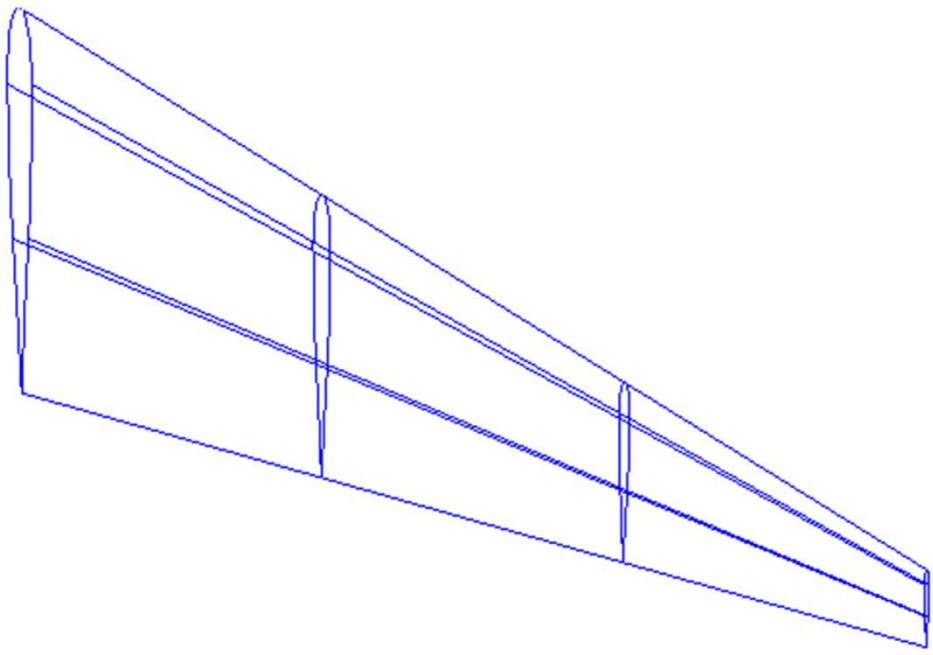
- Semi Span = Semispan
- Root Chord = Cr
- Distance between Ribs = Zm
- Sweep Leading Edge = SweepLE

$$Zm = \frac{Span}{2 \cdot (nRibs - 1)}$$

$$Cr = \frac{2 \cdot Span}{AR \cdot (1 + TRatio)}$$

$$Semispan = \frac{Span}{2}$$

$$SweepLE = atan\left\{tan\left(Sweep25 \cdot \frac{\pi}{180}\right) + \left[\frac{(1 - TRatio)}{AR \cdot (1 + TRatio)}\right]\right\}$$



1.2 Geometrical Definitions

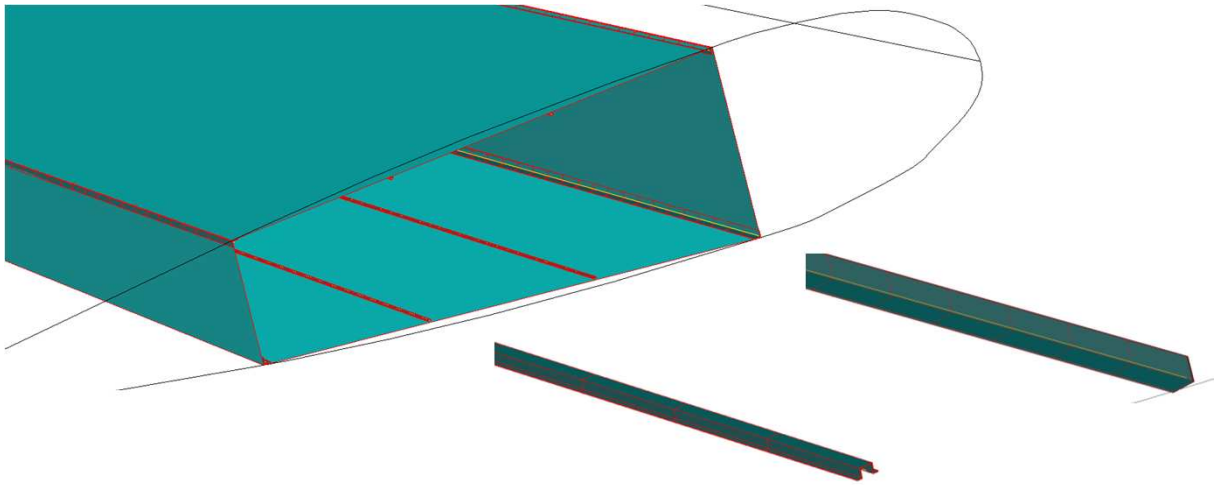
For each new Rib we have a scaling factor, a torsion – that we consider here linear along the span – and a specific origin for the profile in space. This parameters are derived from simple geometrical definitions, where "I" indicates the Rib number (1 for the Root Rib and 4 for the Tip Rib):

$$iScaling = 1 - (1 - TRatio) \cdot \frac{(i-1)}{(nRibs - 1)}$$

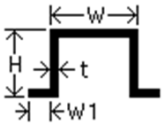
$$iCoord = [(i-1) \cdot Zm \cdot tan(SweepLE) \quad (i-1) \cdot Zm \cdot tan(Dihedral) \quad (i-1) \cdot Zm]$$

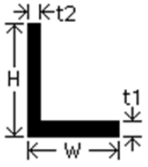
$$iTorsion = \frac{\pi}{180} \cdot \frac{(i-1) \cdot Zm}{(Span/2)} \cdot Torsion$$

For the elements, we chose to use Shell elements for the all skins, ribs and spars, and this property has only thickness as geometrical parameter – for homogeneous isotropic metals. For the bean elements, we chose to use L-Bean strips for the WB corner beans; as for the strips in the WB skin, we chose to use Hat-Bean shaped ones.



L-Bean		Hat-Bean	
H	0.03	H	0.007
W	0.03	t	0.0008
t1	0.002	W	0.01
t2	0.002	W1	0.003

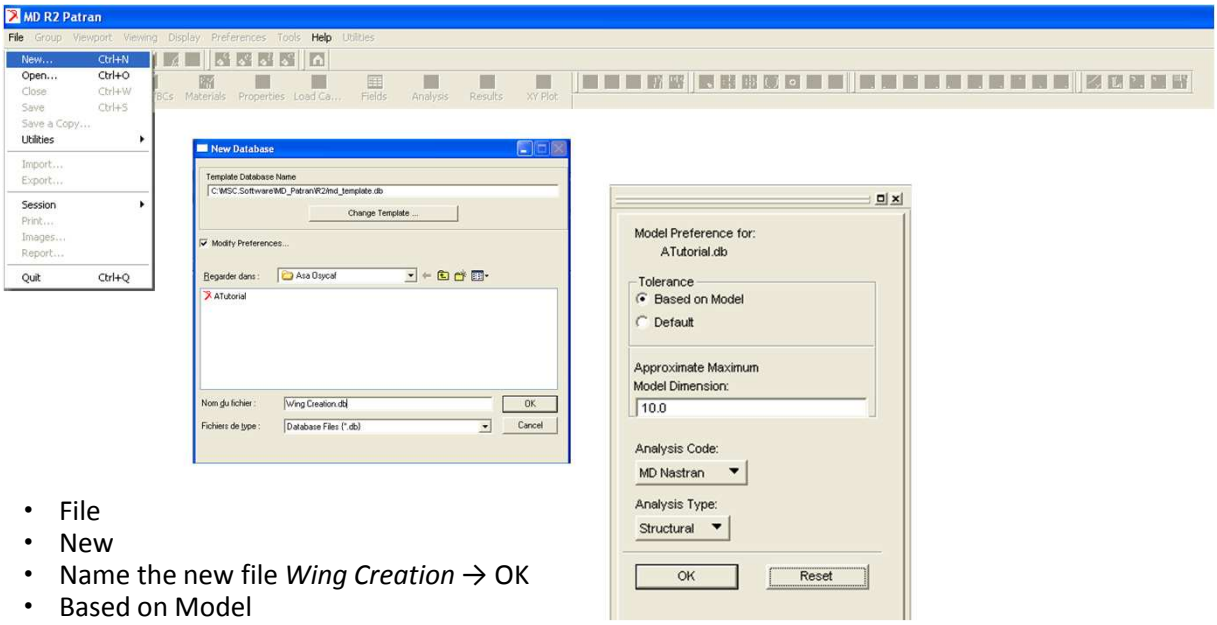




The materials used are all metals with homogeneous and isotropic properties, so Elastic Modulus, Poisson Ratio and density are the only 3 parameters that matter.

	Steel	Aluminium	Titanium	
Elastic Modulus	2,00E+11	6,90E+10	1,20E+11	Pa
Poisson Ratio	0,3	0,3	0,3	-
Density	7800	2770	4110	kg/m3

2.1 Create a New Database



- File
- New
- Name the new file *Wing Creation* → OK
- Based on Model
- Analysis Code: MD Nastran
- Analysis Type: Structural
- OK

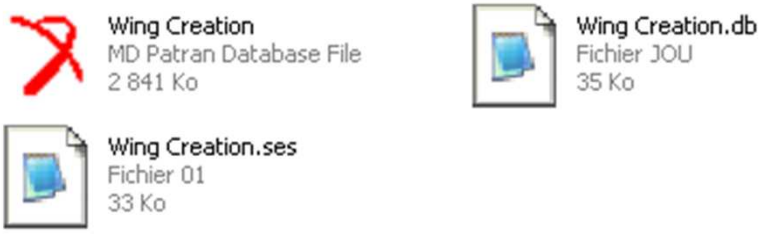
Patran has a toolbar easy to understand and usually intuitive. To construct a model and analyze it the most simple way is to follow the toolbar steps in the correct order, that means:

We start so by creating the *Geometry*. After we should create the *Elements*, that means, mesh the model. In this part we'll need the *Material* and *Properties*, so we do first this two steps and than create the *Elements.* We then create the *Loads/BCs* and start the *Analysis*. *Results* and *XY Plot* are visualization methods, and so the last steps.

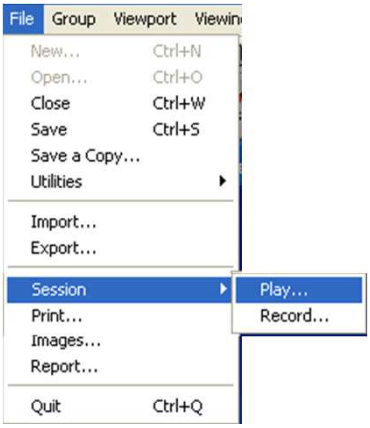
2.2 Patran Files and Sessions

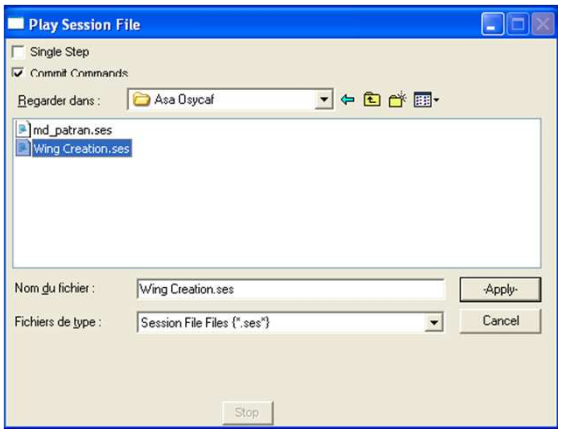
Now go to the folder that Patran register all documents. There you'll find the *Patran Database File* which is the one read by Patran when you open you project or save it. You also have a file *.db*, which is the report that Patran writes every time you create or run a model. This is the file that you'll have to pay attention, because it's inside it that Patran writes the PCL code. So every time you execute an action, go to *Wing Creation.db* and copy the useful code piece and save it in another file, maybe a *.txt.*

Now that if you have the PCL code for a particular action you can modify it and replicate it just by changing it's internal parameters, as will be shown in this tutorial. Also create a file *.ses*, because is this type of file that Patran reads if you play a session, so if you want to read your PCL code, you'll have to copy and paste it inside this *Wing Creation.ses.*



To read a *.ses* file, close all projects and create a new one, as was shown before.





- File
- Session
- Play
- Load the *Wing Creation.ses*
- Apply

After that, Patran will read the PCL code and create everything inside it. Do this activity frequently: it makes easier to find errors inside de code. Note that lines such as "**$#** Point 1 already exists at the…" are just comments and so not important, but "**$?** YESFORALL 26002142" are answers lines and so really important for the automatization of your PCL code.

Patran → file.db → importante piece of code → file.t xt → PCL code → file.ses → Patran . . .

2.3 Creating Groups

It is also important to create groups to better visualize the model, or even to work in a cleaner screen. To create a group, go to the menu and press *group* → *create*.

- Create
- Select Entity
- Give a name for this group
- Make Current
- Unpost All other Groups
- Select all curves, points, elements, nodes, surfaces or any other entity you want in this group
- Apply

- To visualize any group go to the Post option and select the group to be visualized

Note that everything you create will be stored in the current group. Create a group whenever you want.

```
sys_poll_option( 2 )
ga_group_create( “Group Name" )
ga_group_entity_add( " Group Name ", “Entity" )

sys_poll_option( 0 )
uil_viewport_post_groups.posted_groups( "default_viewport", 1, [" Group Name "] )

repaint_graphics(  )
```

2.3 Variables Declaration

We must first declare all the variables that we'll use during the wing construction. It's also possible to declare the variable whenever we need them, but if we organize them all together in the beginning of the PCL code, it'll be easier to change the parameter to create a new wing or with different properties or materials. Remember that the objective of this tutorial is to create a parametric wing, that means, easily changed by simple numerical parameters. We must notice that declaring a variable usually implies in a numerical value. If we want to parameterize, we must respect Patran PCL language that requires single apostrophe for variables usage, such as *y = '2*x'*, and not *y = 2*x*.

```
$# ----------------------------------Wing Basic Parameters---------------------------------
REAL Span = 30
REAL AR = 9
REAL TRatio = 0.2
REAL Sweep25 = 25
REAL Dihedral = 6
REAL Torsion = -3
$# ----------------------------------------Materials-----------------------------------------
REAL Ealum = 69e9
REAL nialum = 0.3
REAL rhoalum = 2770

REAL Esteel = 200e9
REAL nisteel = 0.3
REAL rhosteel = 7800

REAL Etitanium = 120e9
REAL nititanium = 0.3
REAL rhotitanium = 4110
$# ---------------------------------L and Hat Bean Properties--------------------------------
REAL HL = 0.03
REAL WL = 0.03
REAL t1L = 0.002
REAL t2L = 0.002
REAL offL = 0.01

REAL Hhat = 0.007
REAL that = 0.0008
REAL What = 0.01
REAL W1hat = 0.003
REAL offhat = 0.009708

REAL skinthick = 0.002
REAL WBskinthick = 0.002
REAL ribsthick = 0.005
REAL sparsthick = 0.01
$# ---------------------------------Parametric variables-------------------------------------
REAL nRibs = 4
REAL Semispan = `Span/2`
REAL Cr = `2*Span/AR/(1+TRatio)`
REAL Zm = `Span/2/(nRibs-1)`
REAL pi = 3.1415926535
```

2.4 Profile.dat

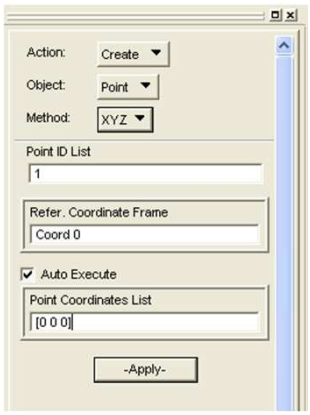
Upper Camber Points:

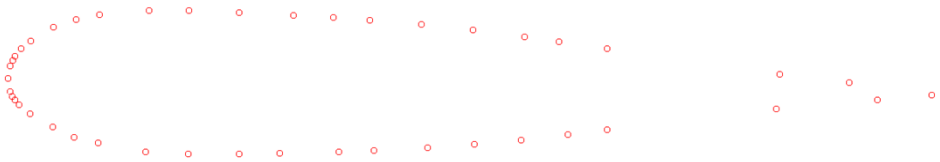
[0.000000 0.017700 0]
[0.002300 0.030900 0]
[0.005000 0.037200 0]
[0.007600 0.041500 0]
[0.014300 0.049900 0]
[0.024900 0.058200 0]
[0.049500 0.073000 0]
[0.074000 0.081400 0]
[0.099000 0.086600 0]
[0.153000 0.090700 0]
[0.196100 0.090500 0]
[0.250400 0.088700 0]
[0.309400 0.085800 0]
[0.352000 0.083300 0]
[0.391900 0.080400 0]
[0.447700 0.075600 0]
[0.503400 0.069600 0]
[0.559300 0.062600 0]
[0.596500 0.057500 0]
[0.648800 0.049800 0]
[0.835100 0.022400 0]
[0.910900 0.013200 0]
[1.000000 0.000300 0]

Lower Camber Points:

[0.002200 0.003800 0]
[0.004900 -0.001800 0]
[0.007200 -0.005300 0]
[0.011900 -0.010600 0]
[0.024300 -0.020400 0]
[0.048600 -0.034200 0]
[0.071600 -0.045700 0]
[0.097900 -0.051600 0]
[0.148800 -0.060700 0]
[0.195300 -0.063200 0]
[0.250100 -0.063200 0]
[0.294500 -0.062600 0]
[0.357900 -0.061000 0]
[0.396500 -0.059500 0]
[0.454300 -0.056300 0]
[0.505000 -0.052700 0]
[0.555600 -0.048200 0]
[0.606300 -0.042700 0]
[0.648500 -0.037500 0]
[0.831700 -0.014900 0]
[0.941000 -0.005300 0]

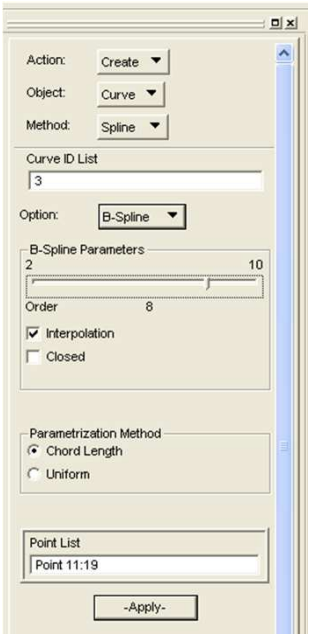
3.1 Creating the Profile

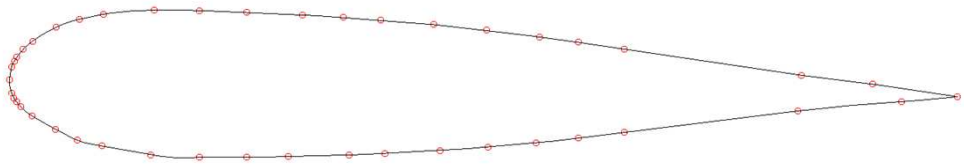




- Create
- Point
- XYZ
- Insert the points from *profile.dat* in the form [coordX coordY coordZ]
- Check if there is no duplicated points. Usually *profile.dat* brings 2 points for the LE and TE. If so, delete one of the duplicated points or be careful not to insert them at first.
- You must have only one point in each position

STRING asm_create_grid_xyz_created_ids[VIRTUAL]
asm_const_grid_xyz("1", "[coordX coordY coordZ]", "Coord 0", asm_create_grid_xyz_created_ids)

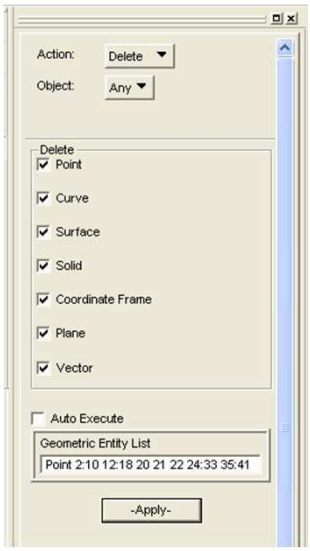


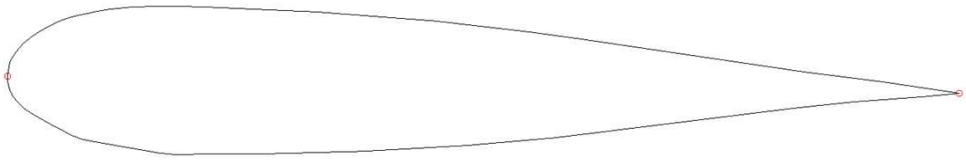


- Create
- Curve
- Spline → B-Spline
- Interpolation (not closed)
- PATRAN accepts no more than 11 points at once for B-Spline, so select a group of consecutive points and B-Spline them. Notice that the order must be equal or lower than the number of points.
- Repeat the B-Spline as many times as necessary to have all the profile curved. Spline the curves from upper and lower camber separately, so that you will have the LE and TE points dividing the profile in 2
- Check if there is no curve deformation and if they match each other in the edge

STRING sgm_curve_bspline_created_ids[VIRTUAL]
sgm_const_curve_bspline("ID", "Point first point:last point", order, TRUE, 1, FALSE, @
sgm_curve_bspline_created_ids)

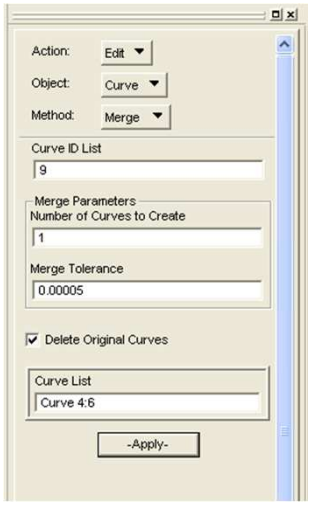
3.1 Creating the Profile

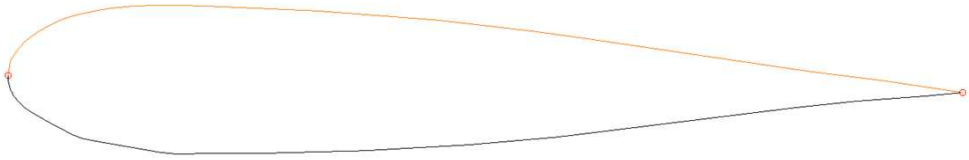




- Delete
- Any
- Select all points but the LE and TE points and delete them

STRING asm_delete_any_deleted_ids[VIRTUAL]
asm_delete_point("Point point", asm_delete_any_deleted_ids)

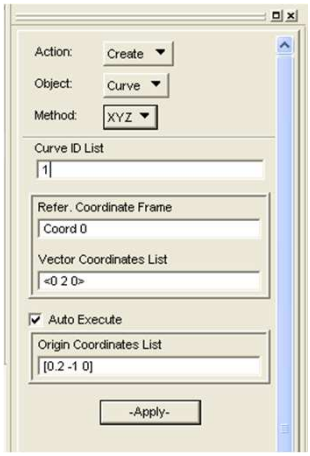




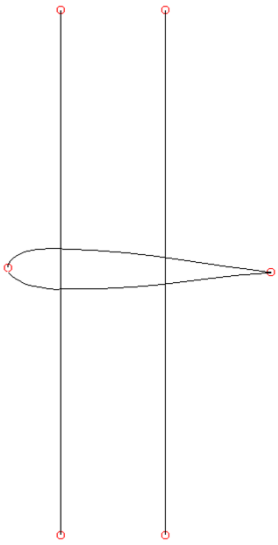
- Edit
- Curve
- Merge
- Set the Merge Tolerance to 0.00005. The default 0.005 is too high
- Delete Original Curves
- Select all curves from upper camber and merge them
- Do the same for the lower camber
- You should have now only 2 curves (lower and upper camber) and 2 points (LE and TE)

STRING sgm_edit_curve_merg_created_ids[VIRTUAL]
sgm_edit_curve_merge(“ID", "Curve first curve:last curve", 1, Merge Tolerance, TRUE, @ sgm_edit_curve_merg_created_ids)

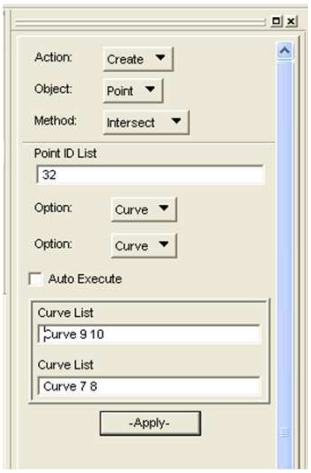
3.2 Creating Wingbox



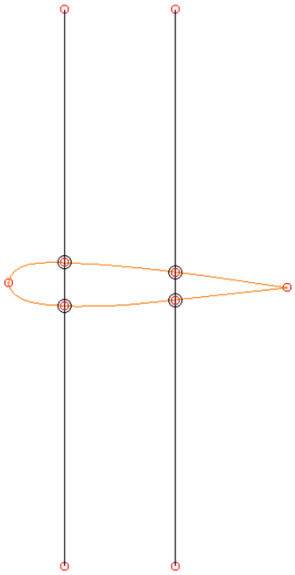
- Create
- Curve
- XYZ
- Create now 2 vertical curves that will limit the Wingbox spars. This 2 curves will be created in 20% and 60% of chord length
- Set a curve of length 2 in y direction <0 2 0>
- Chose the origin to be [0.2 -1 0], so that it will cross the profile in the middle
- Do the same thing for [0.6 -1 0]



STRING asm_create_line_xyz_created_ids[VIRTUAL]
asm_const_line_xyz(“ID", “<vector cood list>", "[origin]", "Coord 0", @ asm_create_line_xyz_created_ids)

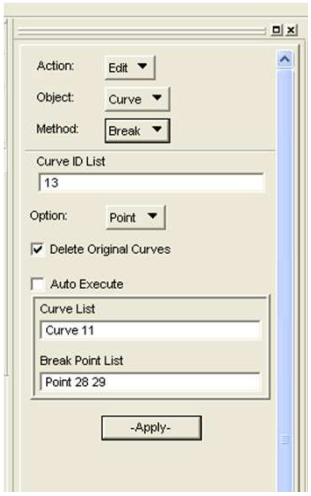


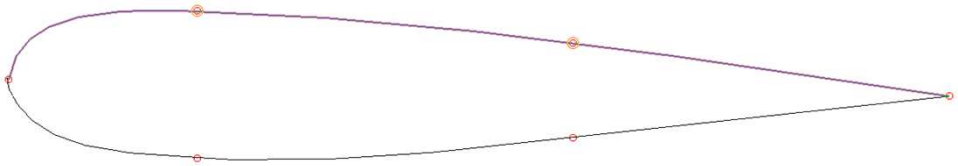
- Create
- Point
- Intersect
- Curve → Curve
- Create 4 points from intersection of the vertical curves and the profile curves
- Be sure that Patran created the correct points and only 4 points
- Then delete the 2 vertical curves



STRING asm_create_grid_int_created_ids[VIRTUAL]
asm_const_grid_intersect_v1(“ID", "Curve 1", "Curve 2", @ asm_create_grid_int_created_ids)

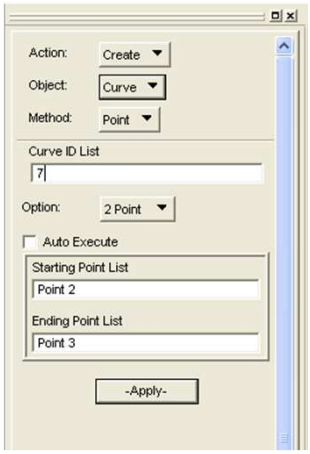
3.2 Creating Wingbox

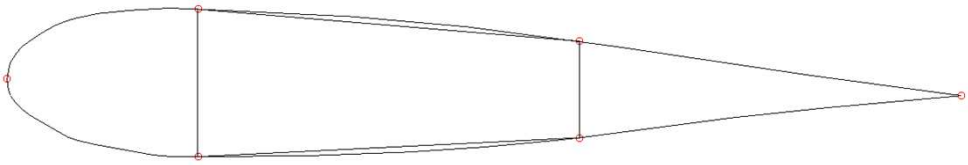




- Edit
- Curve
- Break
- Point
- Delete Original Curves
- Break the upper and lower camber curves by the points just created
- After that you shall have the upper and lower camber divided in 3 curves each. Verify it. You should have 6 curves by now

STRING sgm_curve_break_poi_created_ids[VIRTUAL]
sgm_edit_curve_break_point(“ID", “Point", "Curve", TRUE, @
sgm_curve_break_poi_created_ids)

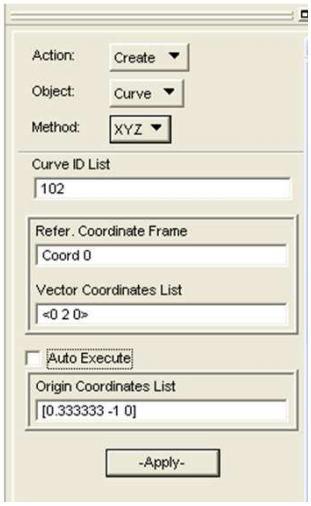




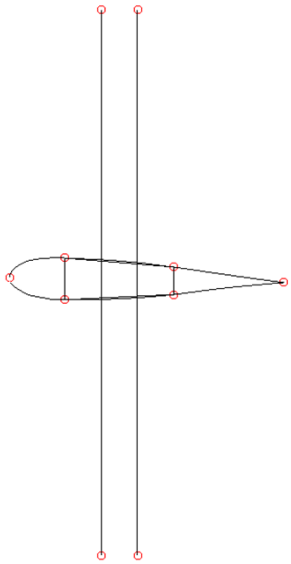
- Create
- Curve
- Point
- Create the upper and lower Wingbox limit curves
- Do the same to create the curves that define the spars
- You should have now a trapezoidal 2D Wingbox and 10 curves

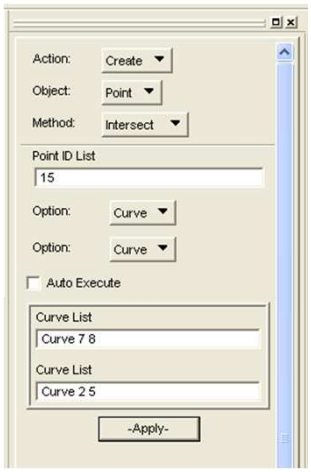
STRING asm_line_2point_created_ids[VIRTUAL]
asm_const_line_2point(“ID", "Point 1", "Point 2", 0, "", 50., 1, @
asm_line_2point_created_ids)

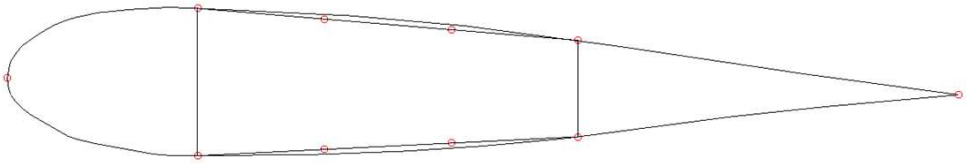
3.2 Creating Wingbox



- Create
- Curve
- XYZ
- Create now 2 vertical curves that will define the strips in the Wingbox. Since the WB has 40% of the chord length, we will put this strips equally spaced.
- Set a curve of length 2 in y direction <0 2 0>
- Chose the origin to be [0.333333 -1 0], so that it will cross the profile 1/3 of WB
- Do the same thing for [0.466667 -1 0]



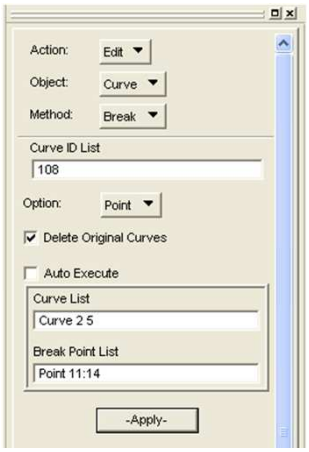


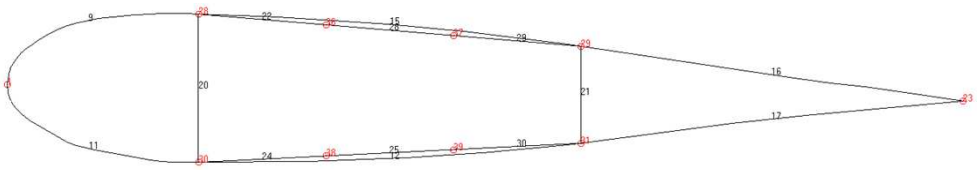


- Create
- Point
- Intersect
- Curve → Curve
- Create 4 points from intersection of the vertical curves and WB skin curves
- Then delete the 2 vertical curves

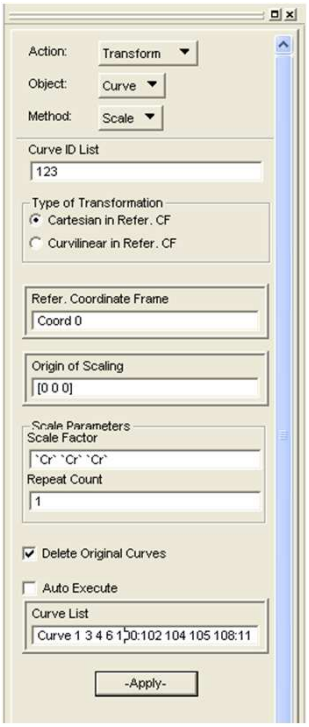
The PCL code for this steps had already been mentioned.

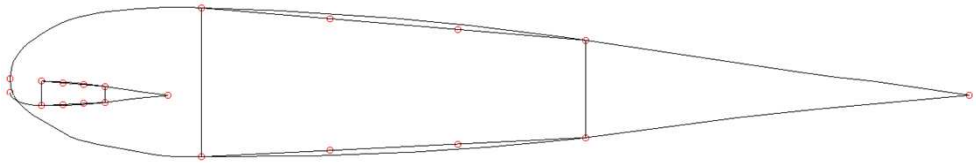
3.2 Creating Wingbox





- Edit
- Curve
- Break
- Break the 2 WB skin curves by the points you just created. Verify if it the curves were really braked
- You should have now a 2D WB with upper and lower skin curves, spars curves and strip points. This totalizes 14 curves

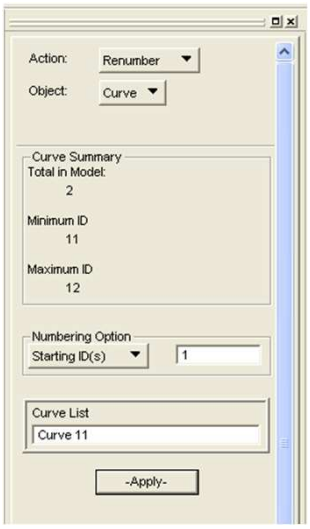


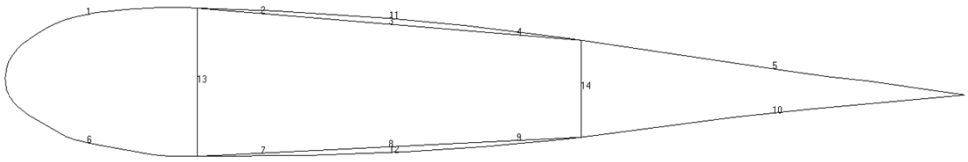


- Transform
- Curve
- Scale
- Set the origin as [0 0 0]
- Up to this moment we had and dimensional wing profile (chord length 1). Set the Scale Factor to the desired chord length `Cr`, in all 3 directions
- Select all curves to do that
- Delete Original Curves and be careful not to delete the new ones!
- Delete now all points of the geometry
- You shall have now only the 14 curves of the scaled profile

STRING sgm_transform_curve_created_ids[VIRTUAL]
sgm_transform_scale(“ID", "curve", [SF SF SF], "[Origin]", “Coord Reference", @
1, TRUE, "Curves", @
sgm_transform_curve_created_ids)

3.3 Creating Wing Frame

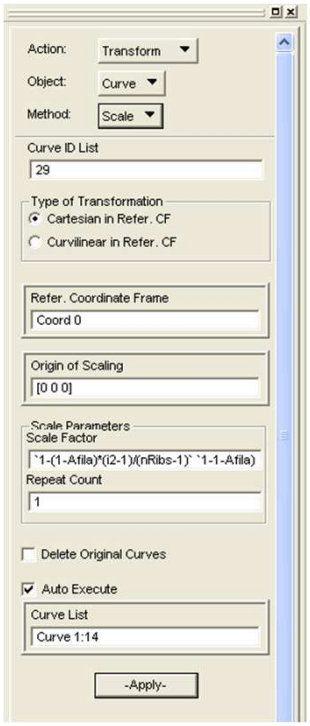


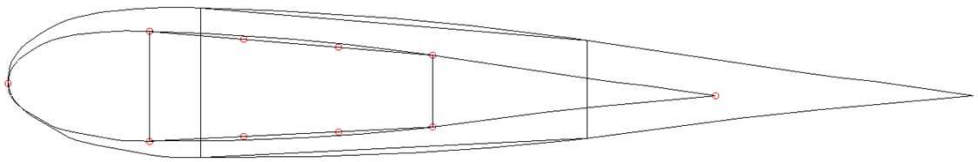


- Renumber
- Curve
- Start the renumbering by ID 1
- Renumber all curves from 1 to 14 as the picture shows
- Renumbering is interesting in this type of construction since when replicating the ribs a logical pattern that will help us construct a great number of geometries and elements using directly the PCL code will be created

- Now you are ready to start the wing construction!

```
STRING sgm_renum_curve_new_ids[VIRTUAL]
sgm_renumber( 1, "curve", “New ID", "Curve to be renumbered", sgm_renum_curve_new_ids )
repaint_graphics( )
```





- Transform
- Curve
- Scale
- Leave the Origin of Scaling in [0 0 0] and Coord 0 as Refer. Coord. Frame
- Since we have 3 more Ribs to build, declare 3 variables `i` so that for each Rib the only parameter changed in the formulas is `i`

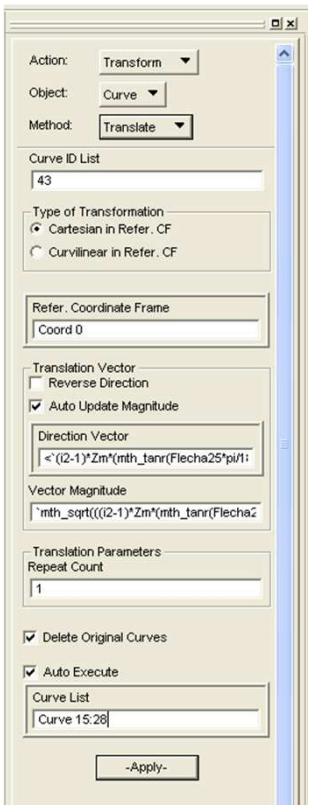
  REAL i2 = 2
  REAL i3 = 3
  REAL i4 = 4
- Set the scale factor to be the one for the second Rib, that means i2 = 2.

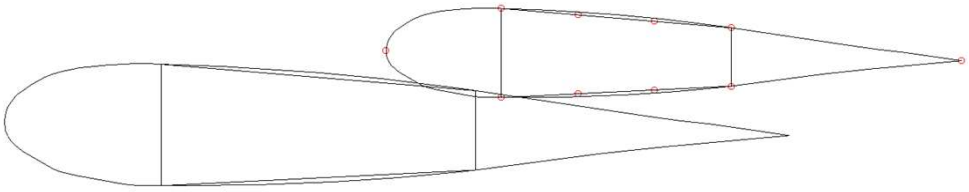
  `1-(1-TRatio)*(i2-1)/(nRibs-1)`
- Note that the scale factor must be repeated in all 3 directions

  `SF` `SF` `SF`
- Don’t delete the original curve, which is Rib 1

```
STRING sgm_transform_curve_created_ids[VIRTUAL]
sgm_transform_scale( “ID", "curve", [Scale Factor], "[0 0 0]",  @
"Coord 0", 1, FALSE, "Curve to be scaled", sgm_transform_curve_created_ids )
```

3.3 Creating Wing Frame





- Transform
- Curve
- Translate
- The direction vector is the same as the new Rib's origin, in the form < >

  <`(i2-1)*Zm*(mth_tanr(Sweep25*pi/180)+(1-TRatio)/(AR*(1+TRatio)))` `(i2-1)*Zm*mth_tand(Dihedral)` `(i2-1)*Zm`>

- The Vector Magnitude is the this same vector's modulus:

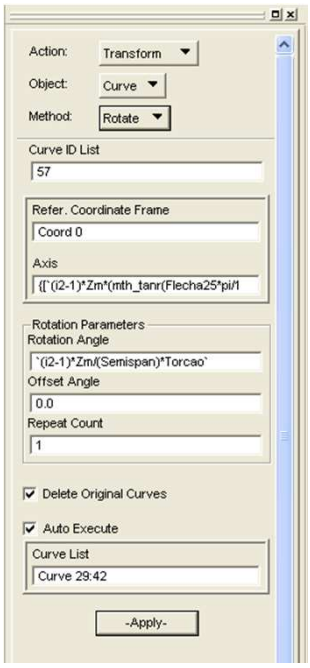
$$|< x\ y\ z >| = \sqrt{x^2 + y^2 + z^2}$$

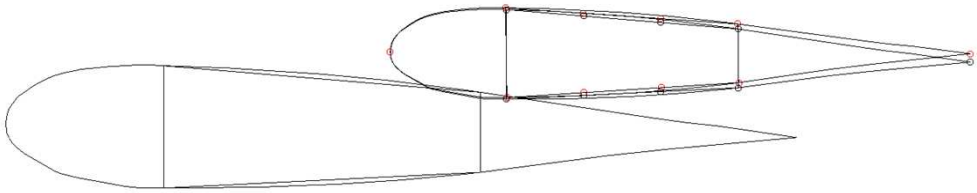
  `mth_sqrt(((i2-1)*Zm*(mth_tanr(Sweep25*pi/180)+(1-TRatio)/(AR*(1+TRatio))))*((i2-1)*Zm*(mth_tanr(Sweep25*pi/180)+(1-TRatio)/(AR*(1+TRatio))))+((i2-1)*Zm*mth_tand(Dihedral))*((i2-1)*Zm*mth_tand(Dihedral))+((i2-1)*Zm)*((i2-1)*Zm))`

- Delete the Original Curves

STRING sgm_transform_curve_created_ids[VIRTUAL]
sgm_transform_translate_v1("ID", "curve", @
"< Direction Vector> Magnitude of the Direction Vector ", FALSE, "Coord 0", 1, @
TRUE, "curves to be translated", sgm_transform_curve_created_ids)

3.3 Creating Wing Frame





- Transform
- Curve
- Rotate
- Note that we must rotate the new Rib around z Axis it's own origin and , not [0 0 0]. The way to define a new Axis is {[New Origin] [Axis Direction]}

  {[`(i2-1)*Zm*(mth_tanr(Sweep25*pi/180)+(1-TRatio)/(AR*(1+TRatio)))` `(i2-1)*Zm*mth_tand(Dihedral)` `(i2-1)*Zm`] [0 0 1]}

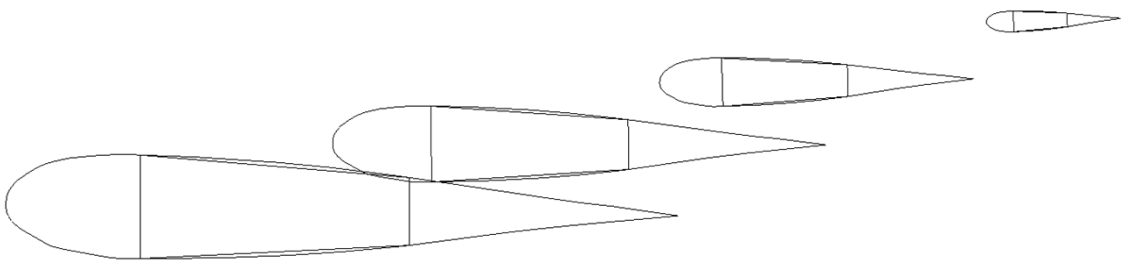
- The Rotation angle is:

  `(i2-1)*Zm/(Semispan)*Torsion`

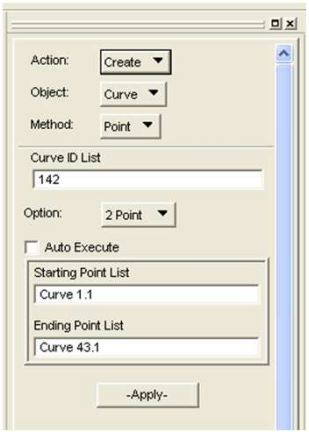
- No offset angle
- Delete Original Curves

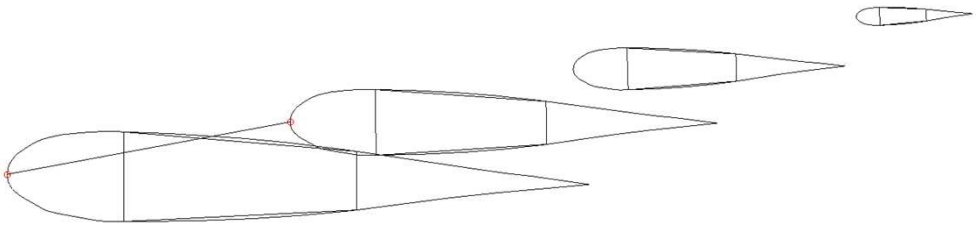
STRING sgm_transform_curve_created_ids[VIRTUAL]
sgm_transform_rotate("ID", "curve", "{[Origin of new Rib] [axis of rotation]}", Rotation angle, 0., "Coord 0", 1, TRUE, @
" curves to be rotated", sgm_transform_curve_created_ids)

By now you have the second Rib built (scaled, translated and rotated). Do the exactly same process to Ribs 3 and 4, that means, i3 and i4. After that you shall have all 4 ribs constructed and ready to be connected by the strips. Delete all points of the geometry because they will not be useful.



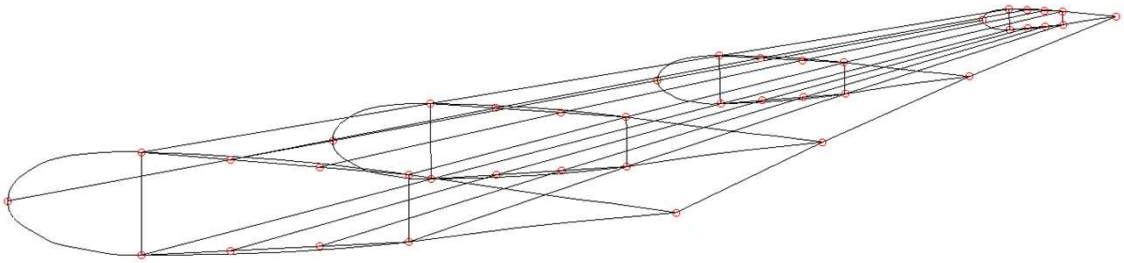
3.3 Creating Wing Frame



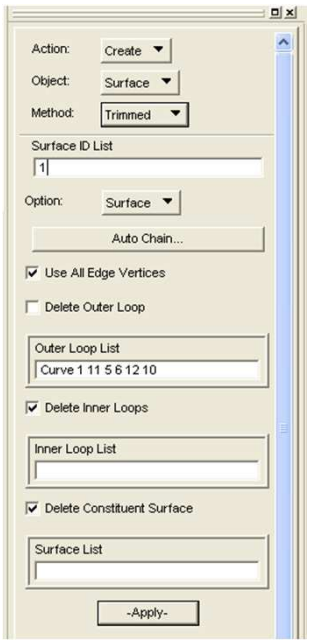


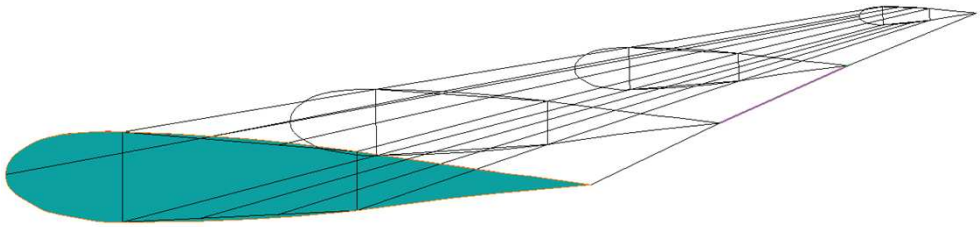
- Create
- Curve
- Point
- 2 Point
- Connect now all similar points to create the frame that will support the strips and surfaces between Ribs.
- Now we don't have points anymore, but we have curve edges. Each curve has 2 edges, so curve N has the following points to be used for a curve creation: Curve N.1 and Curve N.2
- Note that for PCL code we can use the logical pattern between curves from following Ribs to program faster. We can use for this a logical function such as IF function or work with Excel so that we can repeat the PCL code for all different curves faster

By now shall have all curve edges between Ribs connected and your frame should looks like the following picture:



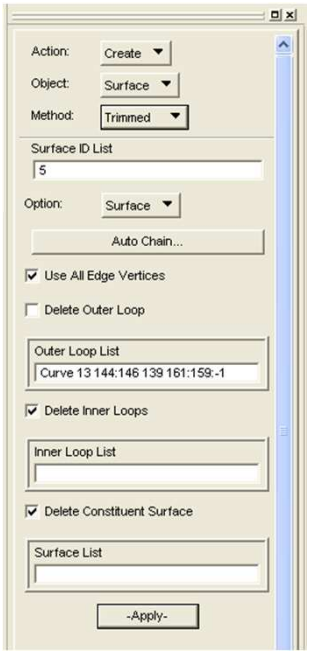
3.4 Creating Surfaces

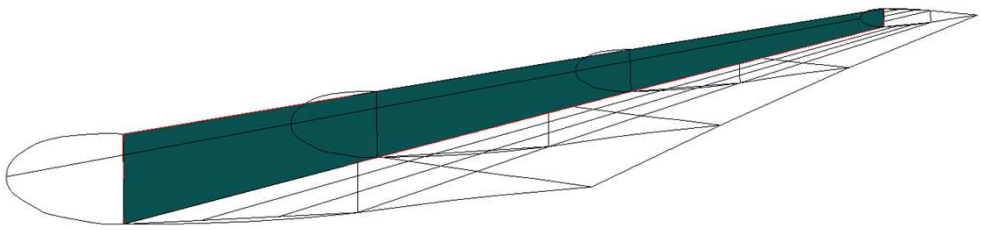




- Create
- Surface
- Trimmed
- Surface
- Use All Edge Vertices
- Don't delete outer loop
- Select the 6 curves that limit the profile
- Repeat this operation to the other 3 Ribs so that we have all first 4 surfaces

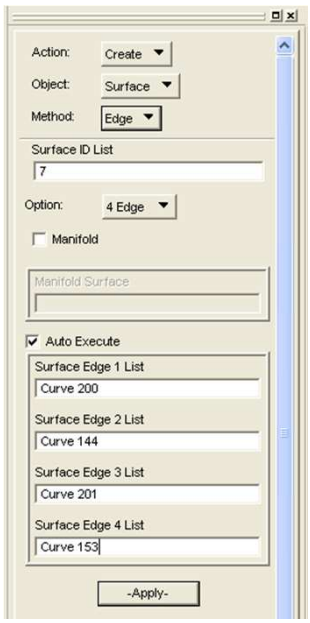
STRING sgm_surface_trimmed__created_id[VIRTUAL]
sgm_create_surface_trimmed_v1("ID", "Curve that define the surface", "", "", @
FALSE, TRUE, TRUE, TRUE, sgm_surface_trimmed__created_id)

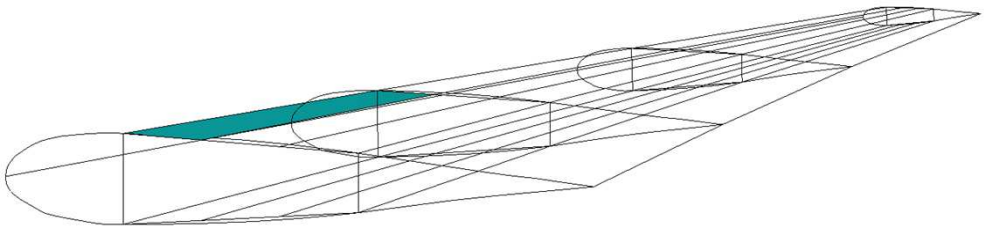




- Create
- Surface
- Trimmed
- Surface
- Use All Edge Vertices
- Don't delete outer loop
- Select now the curves that define the front spar
- Do the same for the back spar so that we create the 2 spar surfaces

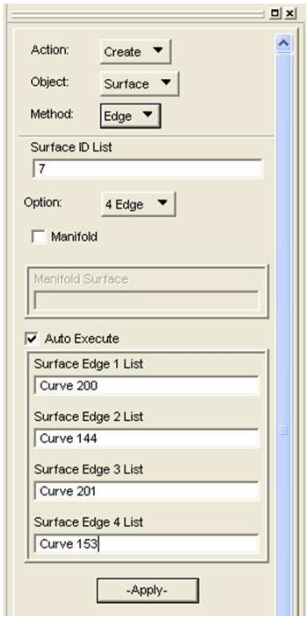
3.4 Creating Surfaces

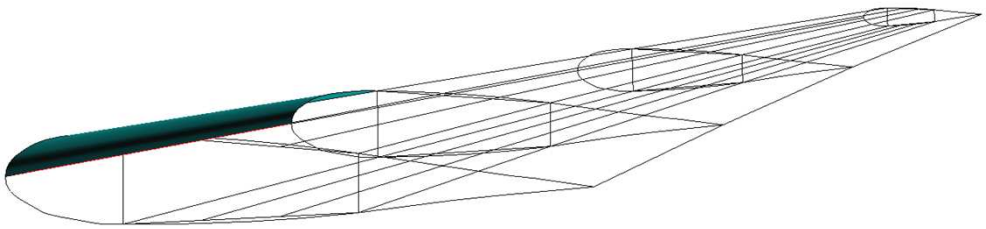




- Create
- Surface
- Edge
- 4 Edge
- Select the 4 curves that limit each skin pannel of the WB
- Start with the WB upper skin. After do the same for the WB lower skin
- You should create 18 surfaces by this method

STRING sgm_surface_4edge_created_ids[VIRTUAL]
sgm_const_surface_4edge(" ID ", "Curve 1 ", "Curve 2 ", "Curve 3 ", @
"Curve 4 ", sgm_surface_4edge_created_ids)



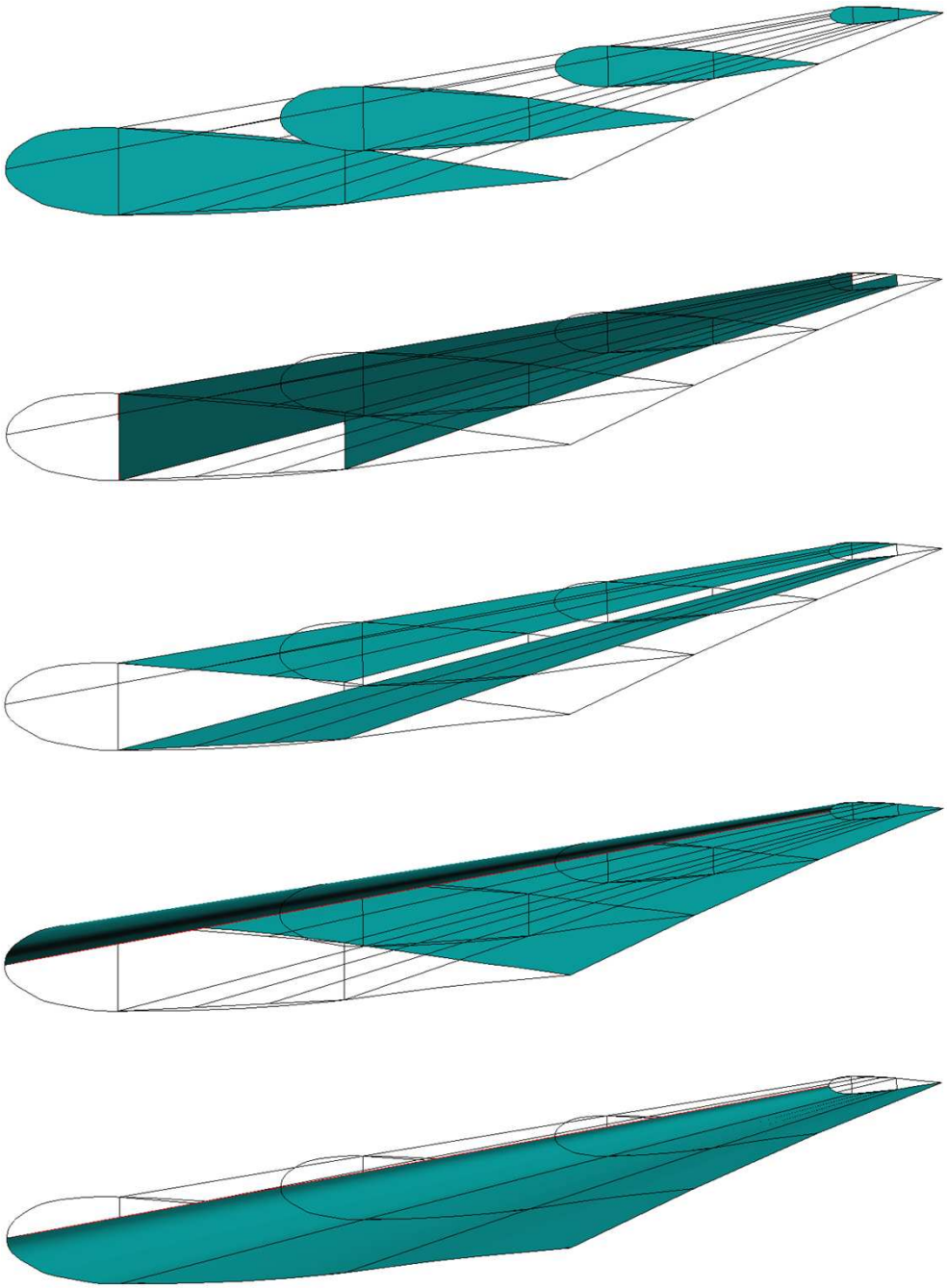


- Create
- Surface
- Edge
- 4 Edge
- Select now the 4 curves that limit each exterior skin pannel of the wing
- Start with the upper camber skin. After do the same for the lower camber skin
- You should also create 18 surfaces this time

STRING sgm_surface_4edge_created_ids[VIRTUAL]
sgm_const_surface_4edge(" ID ", "Curve 1 ", "Curve 2 ", "Curve 3 ", @
"Curve 4 ", sgm_surface_4edge_created_ids)

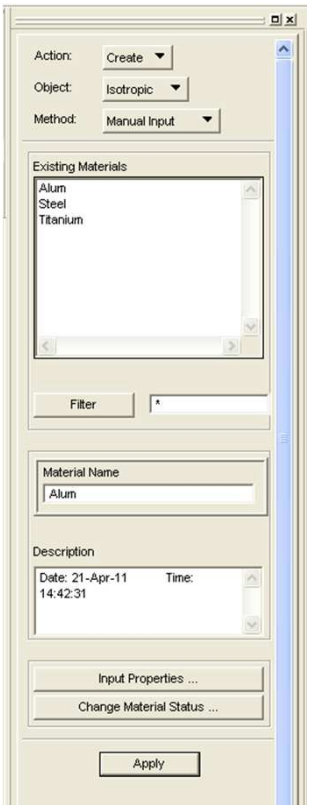
3.4 Creating Surfaces

Now you have all surfaces created and at last you have a entire wing ready to be meshed. But first we'll create all materials and properties.

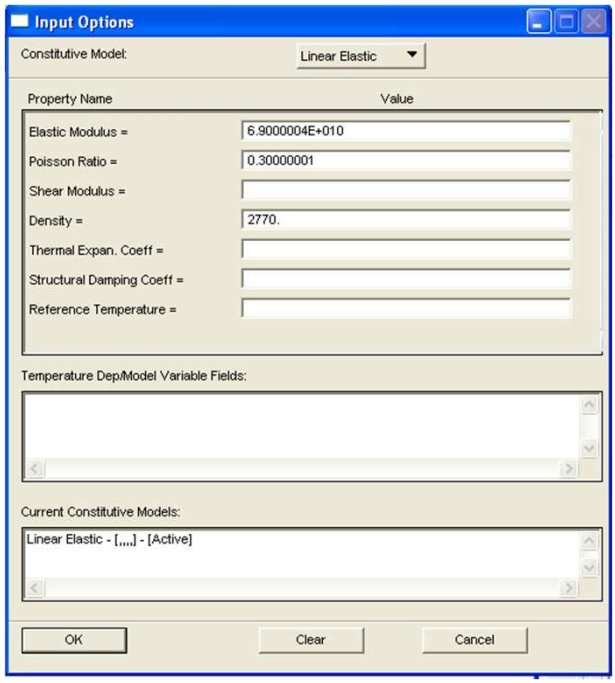


4.1 Creating Materials

To create the materials go to *Materials* in Patran's toolbox. It's the 4th box from left to right.



- Create
- Isotropic
- Manual Input
- Give a name to the material
- Input Properties: input the Elastic Modulus, Poisson Ration and Density
- OK
- Apply

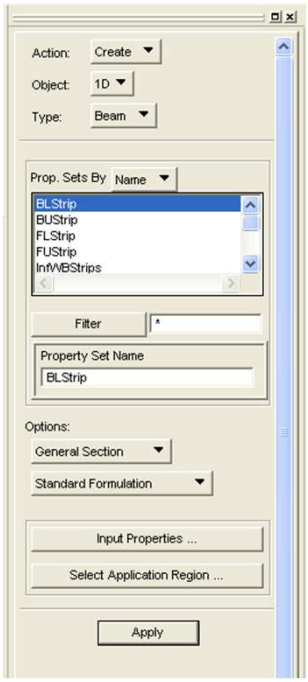


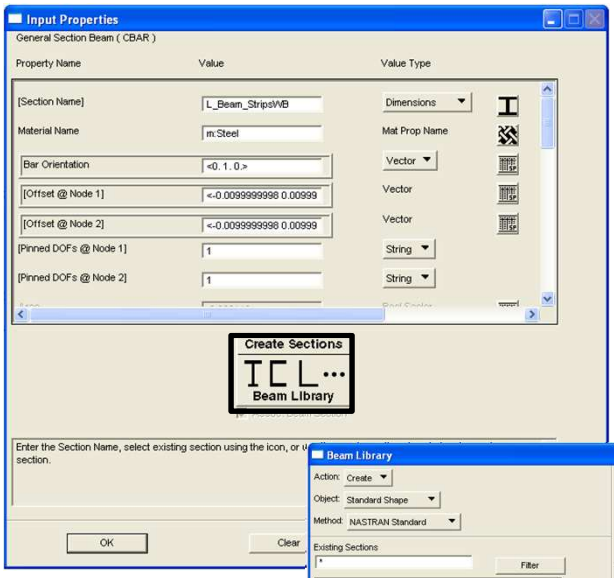
material.create("Analysis code ID", 1, "Analysis type ID", 1, "Name", 0, @
"Date: 14-Apr-11 Time: 16:25:54", "Isotropic", 1, "Directionality", @
1, "Linearity", 1, "Homogeneous", 0, "Linear Elastic", 1, @
"Model Options & IDs", ["", "", "", "", ""], [0, 0, 0, 0, 0], "Active Flag", @
1, "Create", 10, "External Flag", FALSE, "Property IDs", ["Elastic Modulus", @
"Poisson Ratio", "Density"], [2, 5, 16, 0], "Property Values", ["`Elastic Modulus`", "`Poisson Ration`", @
"`Density`", ""])

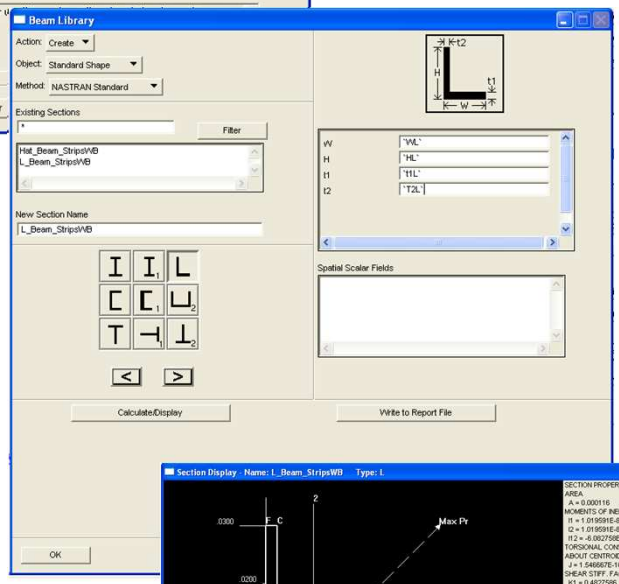
Repeat the same process to create all materials, in this case, Steel, Aluminum and Titanium. Be sure to "apply" every time to create the material. After this, your materials will be saved in the Materials Database and will be ready to be used whenever called.

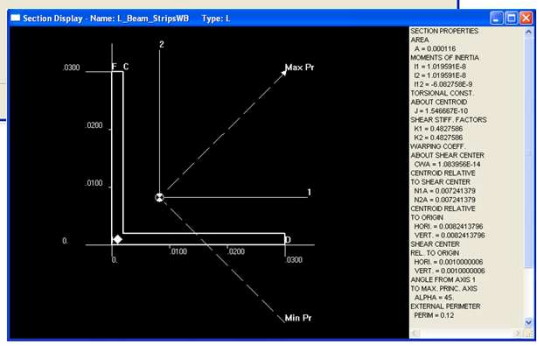
5.1 Creating 1D Properties

To create the properties go to *Properties* in Patran's toolbox. It's the 5th box from left to right. We'll start by creating the 1D elements, that are the strips. We must create the upper and lower WB skin Hat-Bean strips and the 4 WB corner L-Beam strips, so 6 1D properties in total.









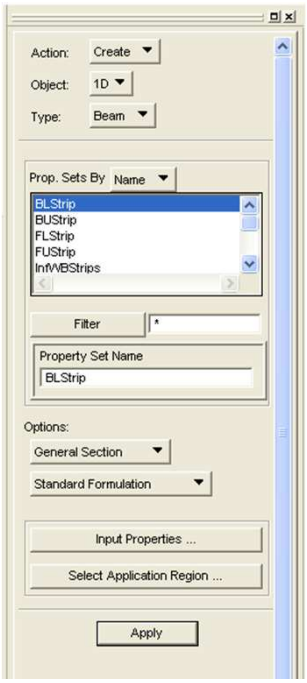
- Create
- 1D
- Beam
- Name the property
- General Section
- Standard Formulation
- Input Properties

- Go to the icon Create Sections
- Name the new section
- Chose a section from the library (L-Beam Section)
- Input all the geometrical parameters. Remember that you have already declared this parameters, so just put them in the ` ` form
- Calculate/Display to see the real geometry numerical parameters
- Get the *offset* values and declare them

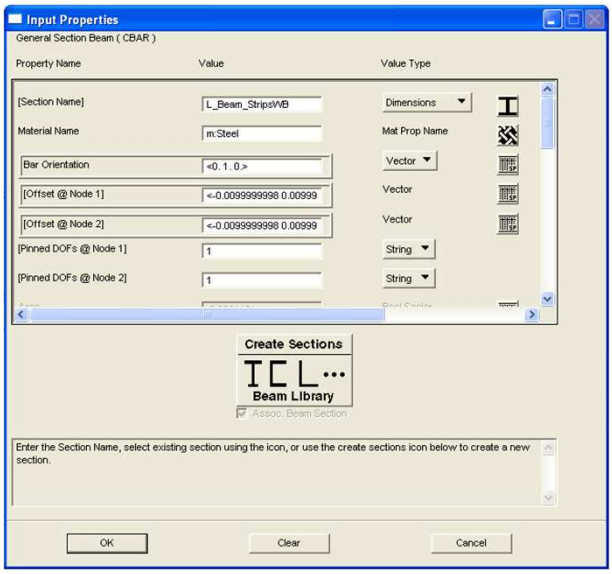
- Do the same for the Hat-Beam and you shall have now 2 Beam Cross Sections defined and ready to be used in the property definition.

beam_section_create("Name", "type", ["`P1`", "`P2`", "`P3`", "`P4`" @
])

5.1 Creating 1D Properties



- Go now to Input Properties again

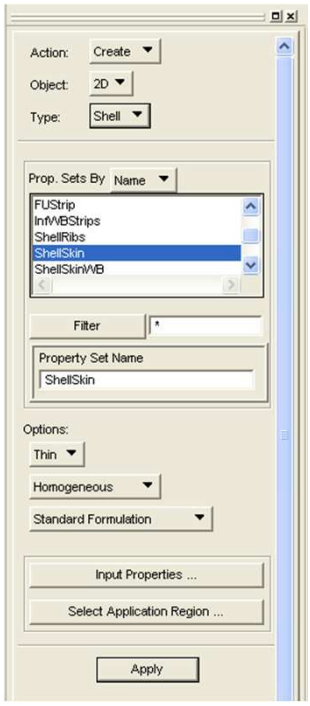


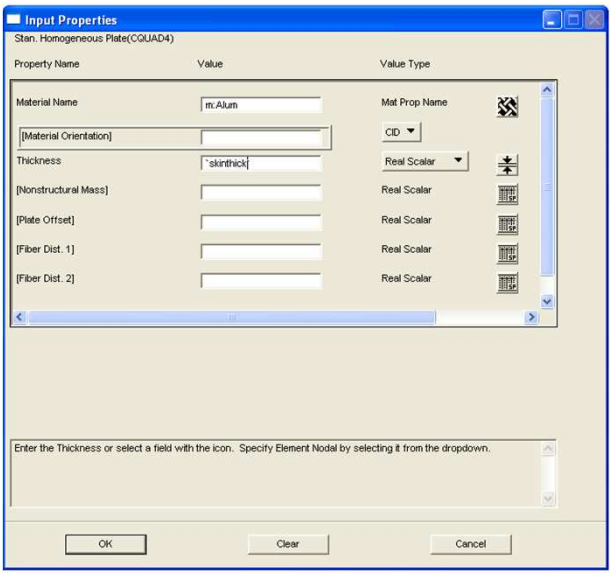
- Chose now the section name from the icon. You must have the L-Beam and Hat-Beam cross sections there
- Chose the material from the icon. You must have Steel, Aluminum and Titanium there
- Now input the Bar Orientation and offsets in the form < > This is important to have the bean section well orientated. For the offsets < + - 0 >, for example, means < `offset` `-offset` 0 >
- Follow this table to create all 1D properties
- Select the Application region according to each property. In this case, this region must be curves.

Property	Section	Material	Bar Orientation	Offset 1	Offset 2	Pinned 1	Pinned 2
FUStrip	L-Beam	Steel	< 0 -1 0 >	< + - 0 >	< + - 0 >	1	1
FLStrip	L-Beam	Steel	< 1 0 0 >	< + + 0>	< + + 0>	1	1
BUStrip	L-Beam	Steel	< -1 0 0 >	< - - 0 >	< - - 0 >	1	1
BLStrip	L-Beam	Steel	< 0 1 0 >	< - + 0 >	< - + 0 >	1	1
SupWBStrips	Hat-Beam	Steel	< 0 1 0 >	< 0 - 0 >	< 0 - 0 >	1	1
InfWBStrips	Hat-Beam	Steel	< 0 -1 0 >	< 0 + 0 >	< 0 + 0 >	1	1

```
elementprops_create( “Name", 11, 2, 42, 1, 1, 20, [39, 13, 6, 4042,  @
4043, 2047, 2048, 1, 10, 11, 4026, 1026, 4044, 4045, 4037, 4047, 4048, 4050,  @
4051, 4053, 4054, 4056, 4057, 4061, 8200, 8201, 8202], [11, 5, 2, 2, 2, 4, 4,  @
1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 6, 4, 4, 4], [ @
“Cross Section Name", "m:Material", "<Orientation>", "<Offset 1>", "<Offset 2>", “Pinned 1", @
 “Pinned 2", "", "", "", "", "", "", "", "", "", "", "", "", "", "", "", "", "", "", "", @
"Analysis", "Analysis", "Analysis"], "" )
```

5.2 Creating 2D Properties





- Create
- 2D
- Shell
- Name the property
- Thin
- Homogeneous
- Standard Formulation
- Input Properties

- Input the material
- Input the Thickness. Note that you have already declared it
- OK

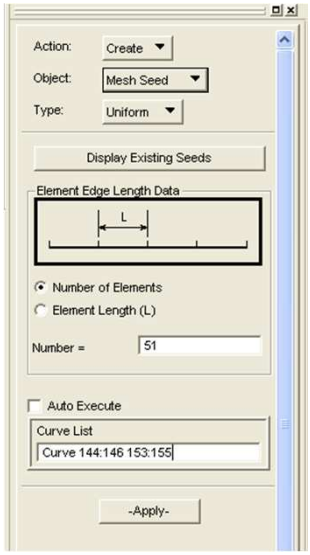
Do the same for the Skin Shell in Aluminum, the WB Skin Shell in Aluminum and the Ribs Shell in Aluminum and the Spars Shell in Titanium. For 2D properties the application region must be a surface.

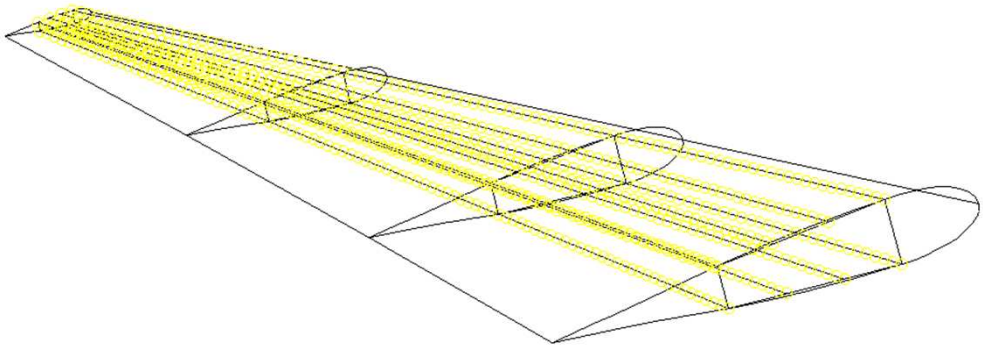
elementprops_create(“Name", 51, 25, 35, 1, 1, 20, [13, 20, 36, 4037, @
4111, 4118, 4119], [5, 9, 1, 1, 1, 1, 1], ["m:Material", "", "`thickness`", "", "", @
"", ""], “Surfaces from Application Region")

By now we have a wing ready to be meshed!

6.1 Creating Meshing Seed

We start now to mesh the wing. But to before this, we should define a Mesh Seed for some geometries so we can have a refined mesh were it is necessary. To create the mesh go to *Elements* in Patran's toolbox. It's the 2nd box from left to right.

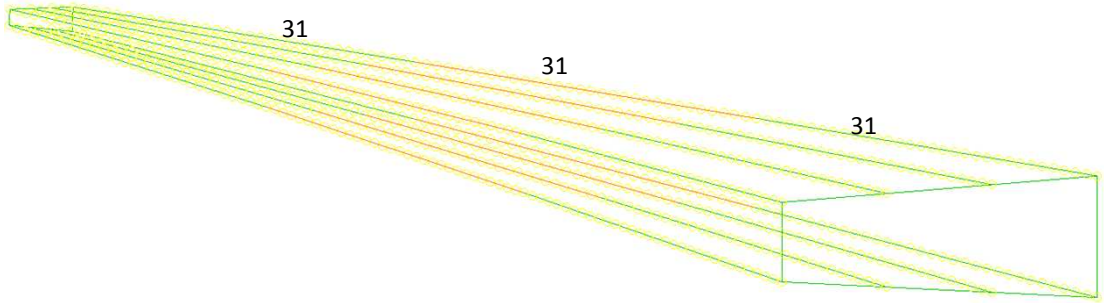




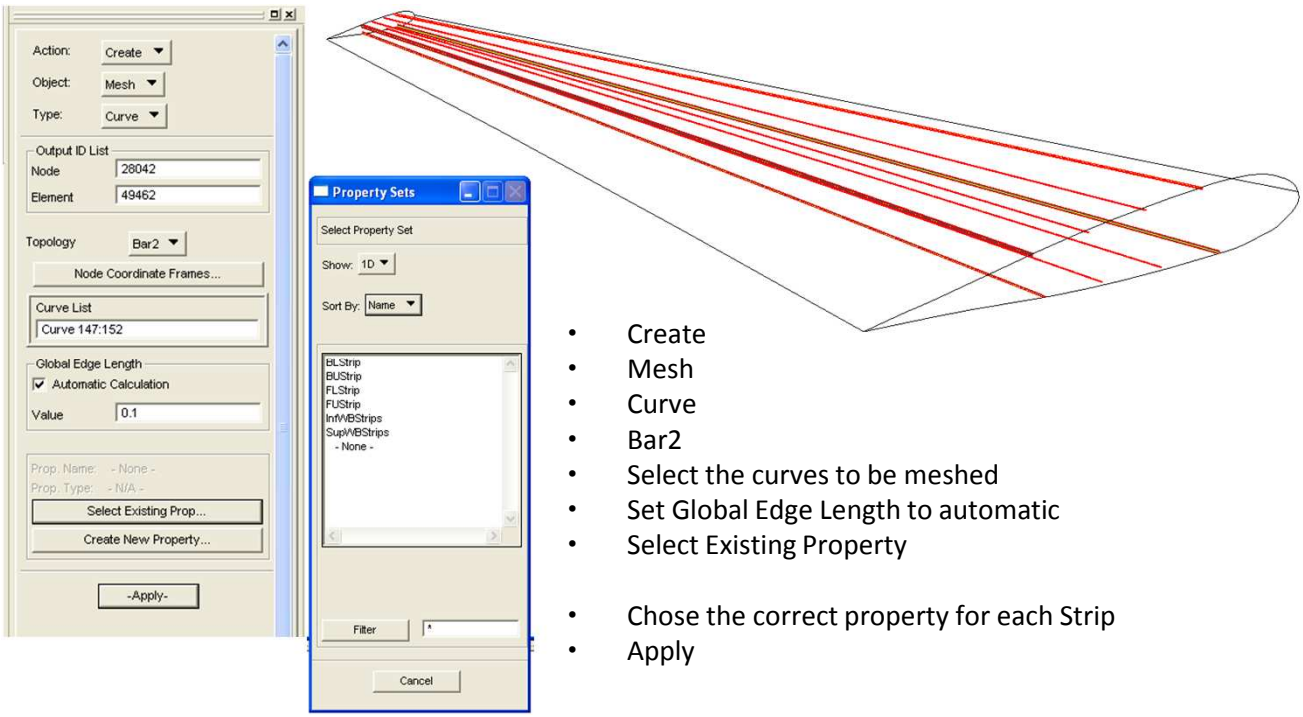
- Number of Elements
- Insert the number of mesh seeds you wish for a specific curve
- Select the curves
- Apply

ui_exec_function("mesh_seed_display_mgr", "init")
mesh_seed_create("Curve to receive Mesh Seed", 1, Number of Mesh Seed, 0., 0., 0.)

Repeat this operation to all edge curves that define the Wingbox. Use the Mesh Seed definition indicated in the picture below, that means, 31 seeds for each curve that composes the WB strips. Note that we have 8 srips, and each one is formed by 3 curves, so we'll have 24 curves with 31 seeds each



6.2 Meshing Strips

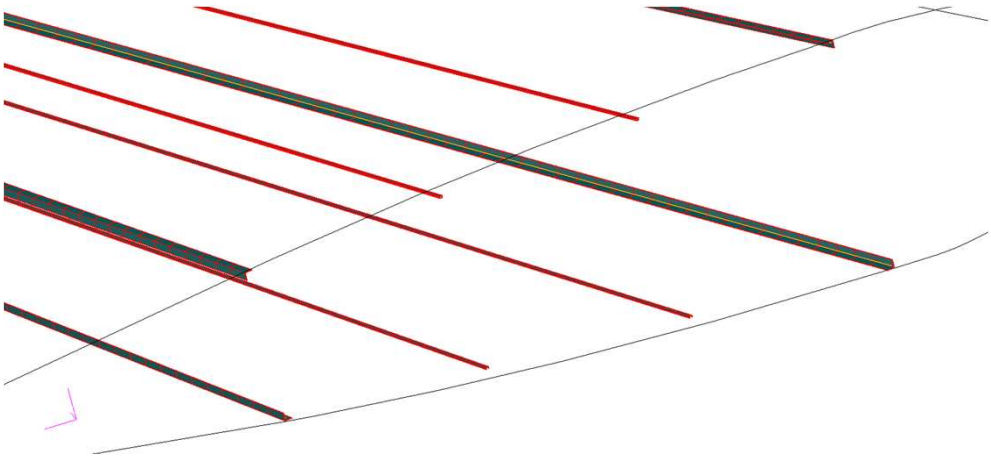


- Create
- Mesh
- Curve
- Bar2
- Select the curves to be meshed
- Set Global Edge Length to automatic
- Select Existing Property

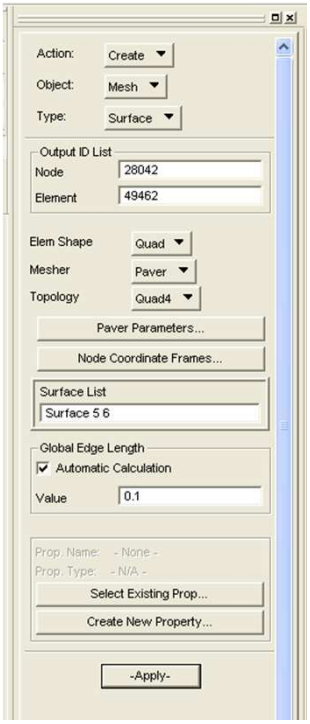
- Chose the correct property for each Strip
- Apply

ui_exec_function("mesh_seed_display_mgr", "init")
INTEGER fem_create_mesh_curve_num_nodes
INTEGER fem_create_mesh_curve_num_elems
STRING fem_create_mesh_c_nodes_created[VIRTUAL]
STRING fem_create_mesh_c_elems_created[VIRTUAL]
fem_create_mesh_curv_1(“Curves to be Meshed", 16384, length, "Bar2", "#", "#", @
"Coord 0", "Coord 0", fem_create_mesh_curve_num_nodes, @
fem_create_mesh_curve_num_elems, fem_create_mesh_c_nodes_created, @
fem_create_mesh_c_elems_created)
fem_associate_elems_to_ep(“Property", “Elements to associate this property", number of associations)

Repeat the same operation for all strips and you shall arrive in well orientated beans as shows the picture below.



6.3 Meshing Surfaces



- Create
- Mesh
- Surface
- Quad or Tria
- Paver or IsoMesh
- Quad4 or Tria3
- Select the Surfaces to be meshed
- Set Global Edge Length to automatic
- Select Existing Property

- Chose the correct property for each surface
- Apply

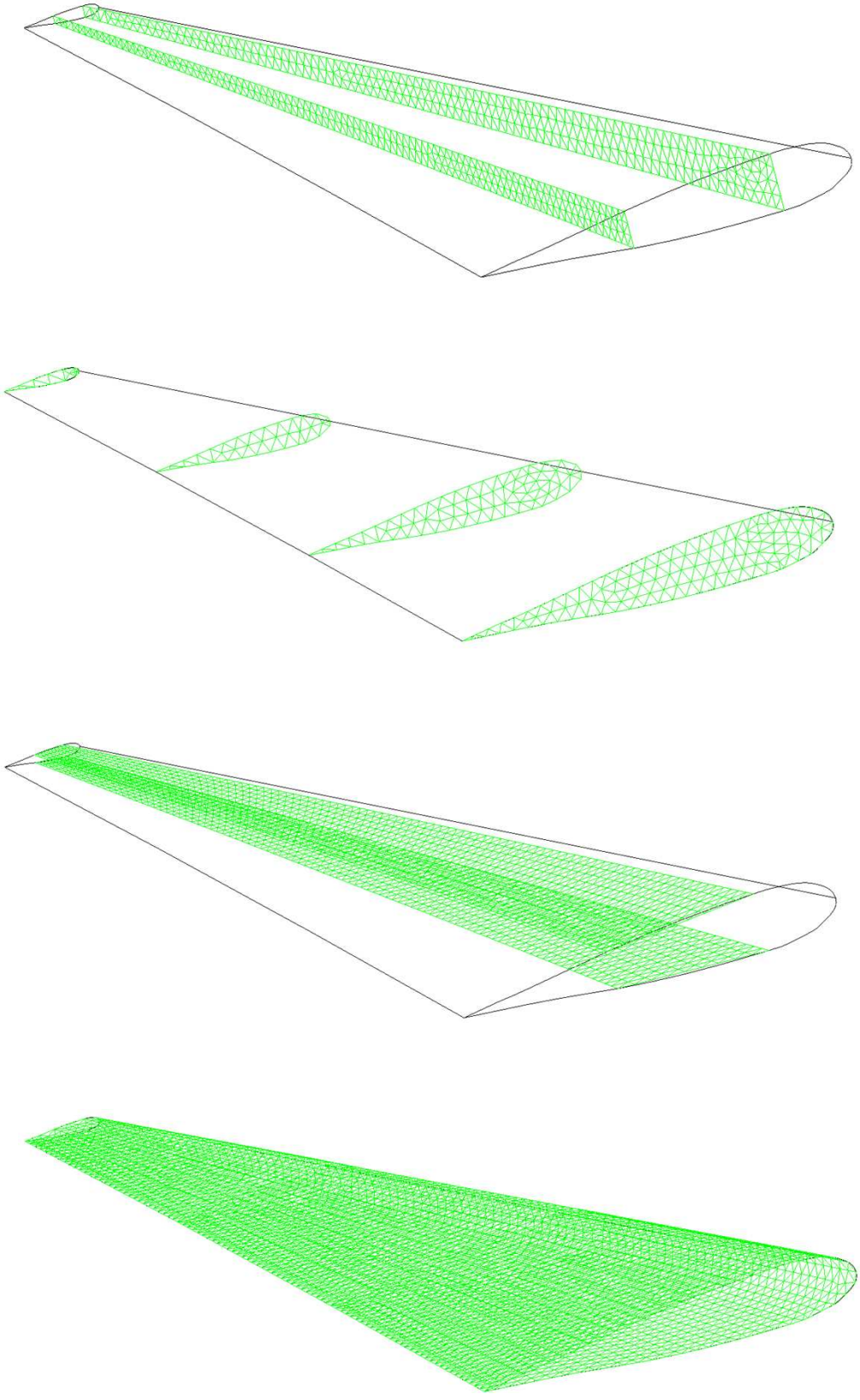
ui_exec_function("mesh_seed_display_mgr", "init")
INTEGER fem_create_mesh_surfa_num_nodes
INTEGER fem_create_mesh_surfa_num_elems
STRING fem_create_mesh_s_nodes_created[VIRTUAL]
STRING fem_create_mesh_s_elems_created[VIRTUAL]
fem_create_mesh_surf_4(“Mesher", 49680, "Surfaces to be meshed", 4, [“GEL", " GEL ", " GEL " @
, " GEL "], “Topology", "#", "#", "Coord 0", "Coord 0", @
fem_create_mesh_surfa_num_nodes, fem_create_mesh_surfa_num_elems, @
fem_create_mesh_s_nodes_created, fem_create_mesh_s_elems_created)
fem_associate_elems_to_ep(“Property", “Elements to associate this property", number of associations)

Do this step for all group of surfaces: Spars, Ribs, WB Skin, Upper Camber Skin and Lower Camber Skin. The type of mesh and Global Edge Length for each surface is in the table below:

Surface	Element Shape	Mesher	Topology	Global Edge Length
Spars	Tria	Paver	Tria3	Auto
Ribs	Tria	Paver	Tria3	Auto
WB Skin	Quad	IsoMesh	Quad4	Auto
Upper C. Skin	Quad	IsoMesh	Quad4	Auto
Lower C. Skin	Quad	IsoMesh	Quad4	Auto

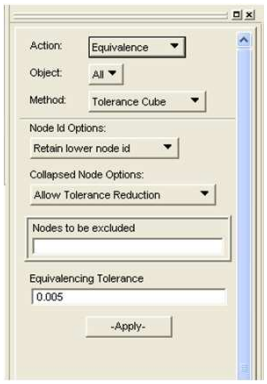
6.3 Meshing Surfaces

After meshing all surfaces you must have the following mesh configuration:



7.1 Verifying Equivalence

Now that all meshing is complete we must verify if there are equivalence between nodes, and if so, delete one of them.



- Equivalence
- All
- Tolerance Cube
- Retain lower node id
- Allow tolerance reduction
- Set the Equivalencing Tolerance to 0.005, and if necessary, Patran will reduce it automatically
- Apply

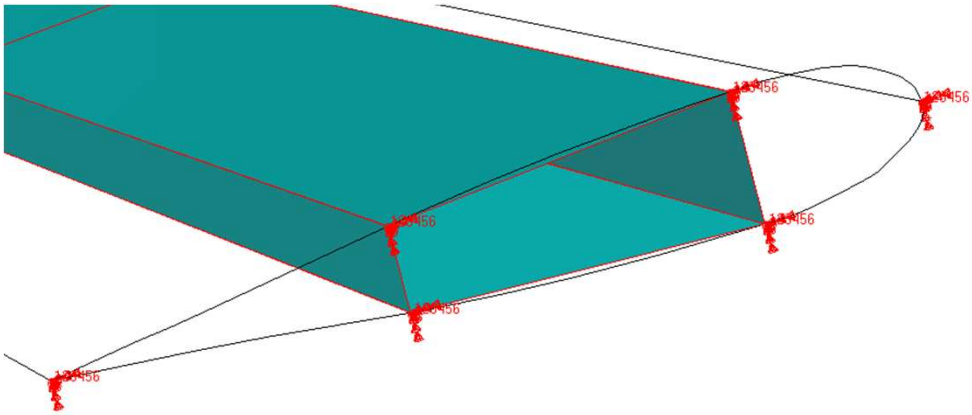
- Same nodes will be deleted in this step

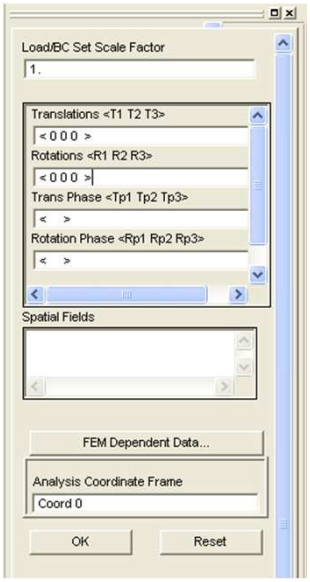
```
ui_exec_function( "mesh_seed_display_mgr", "init" )
verify_boundaries_display_mgr.initialize(  )
REAL fem_equiv_all_x_equivtol_ab
INTEGER fem_equiv_all_x_segment
fem_equiv_all_group4( [" "], 0, "", 1, 1, 0, Equivalencing Tolerance, FALSE,  @
fem_equiv_all_x_equivtol_ab, fem_equiv_all_x_segment )
repaint_graphics(  )
```

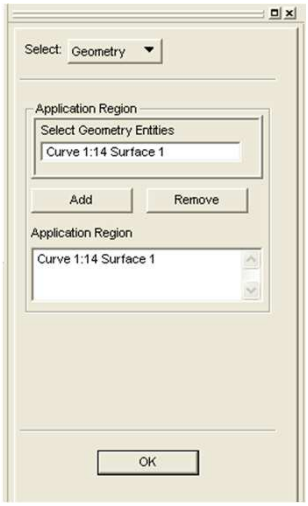
By now we have a wing ready to receive the boundary conditions and loadings!

8.1 Creating Boundary Conditions for a Cantilever Wing

To create the boundary conditions and loads go to *Loads/BCs* in Patran's toolbox. It's the 3rd box from left to right. We'll start by creating fixation of the wing, and since it is a cantilever wing, we must restrict all translation and rotation in it's root.







- Create
- Displacement
- Nodal
- Default
- Set the name as Fix
- Input Data…

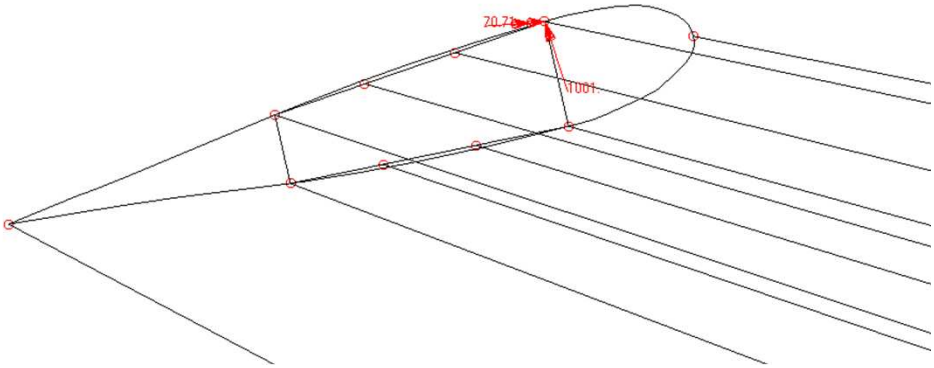
- Set < 0 0 0 > for both translation and rotation degrees of freedom
- OK

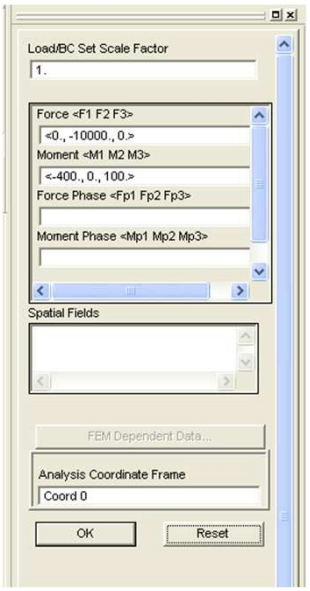
- Select Application Region…
- Geometry
- Add the first Rib surface
- OK

- Apply

loadsbcs_create2("Name", "Displacement", "Nodal", "", "Static", [@
"Application Region"], "Geometry", "Coord 0", "1.", ["< T1 T2 T3 >", @
"< R1 R2 R3 >", "< >", "< >"], ["", "", "", ""])

8.2 Creating Forces





- Create
- Force
- Nodal
- Default
- Set the name as Force
- Input Data…

- Set < 50 1000 0 > for Force and < -50 0 -50 > for the Moment
- Note that we defined everything in SI, so force and moment here must be in N and N.m
- OK

- Select Application Region…
- Geometry
- Add the point as the picture shows
- OK

- Apply

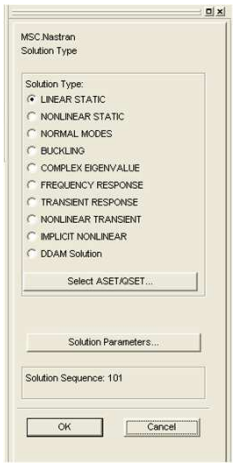
loadsbcs_create2(“Name", "Force", "Nodal", "", "Static", [“Application Region"], @
"Geometry", "Coord 0", "1.", ["< Fx Fy Fz >", "< Mx My Mz >", @
"< >", "< >"], ["", "", "", ""])

9.1 Analyze Model in Nastran and Results: SOL101 Linear Static

To analyze the model go to *Analyze* in Patran's toolbox. It's the 3rd box from right to left. We'll start by analyzing the model by LINEAR STATIC Solution (SOL101). When Analysis starts Nastran will be opened and will close after a sound beep. A *.bdf* file will be created (look for file F06). Search for "FATAL" in this file and make fix the problem until any *fatal error* exists anymore.

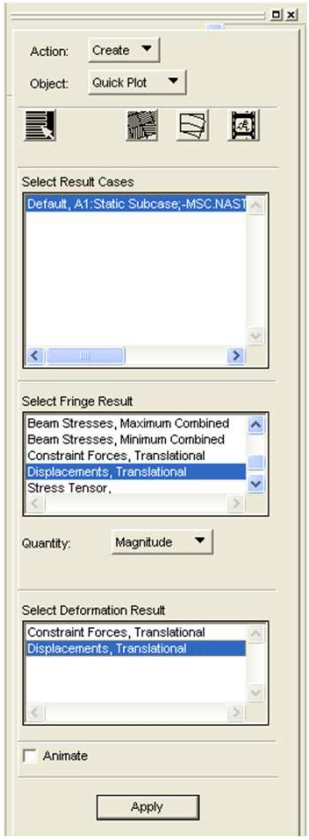
- Analyze
- Entire Model
- Full Run

- Go to Solution Type…
- Select LINEAR STATIC (SOL101)
- Ok
- Apply



- Access Results
- Attach XDB
- Results Entities
- Select Result File

- Select the correct XDB file
- Apply

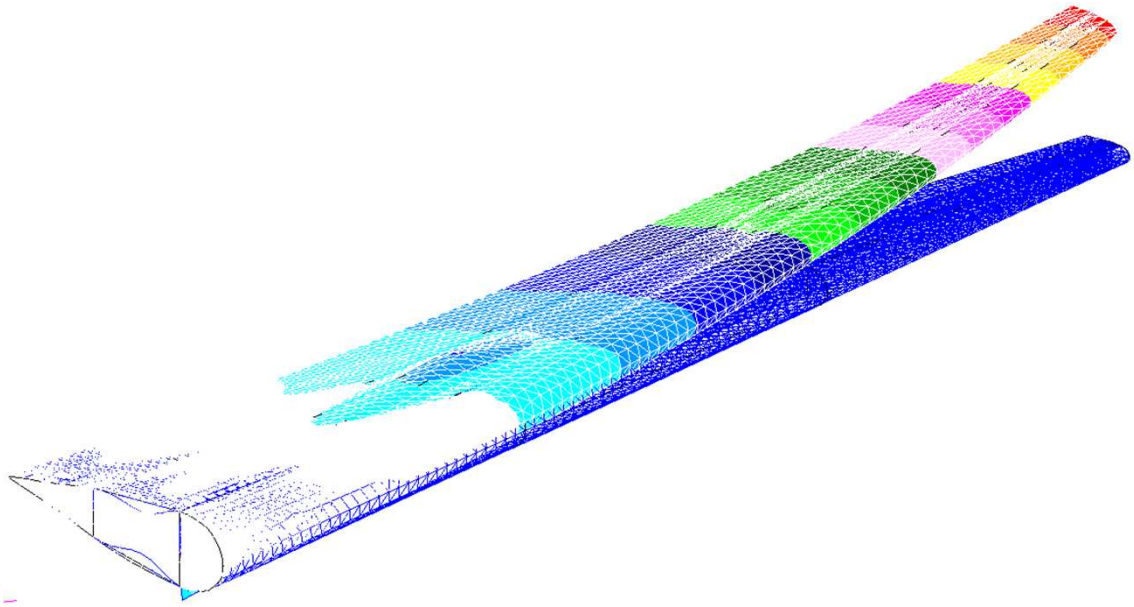


To visualize the results from Nastran analysis go to *Results* in Patran's toolbox. It's the 2nd box from right to left.

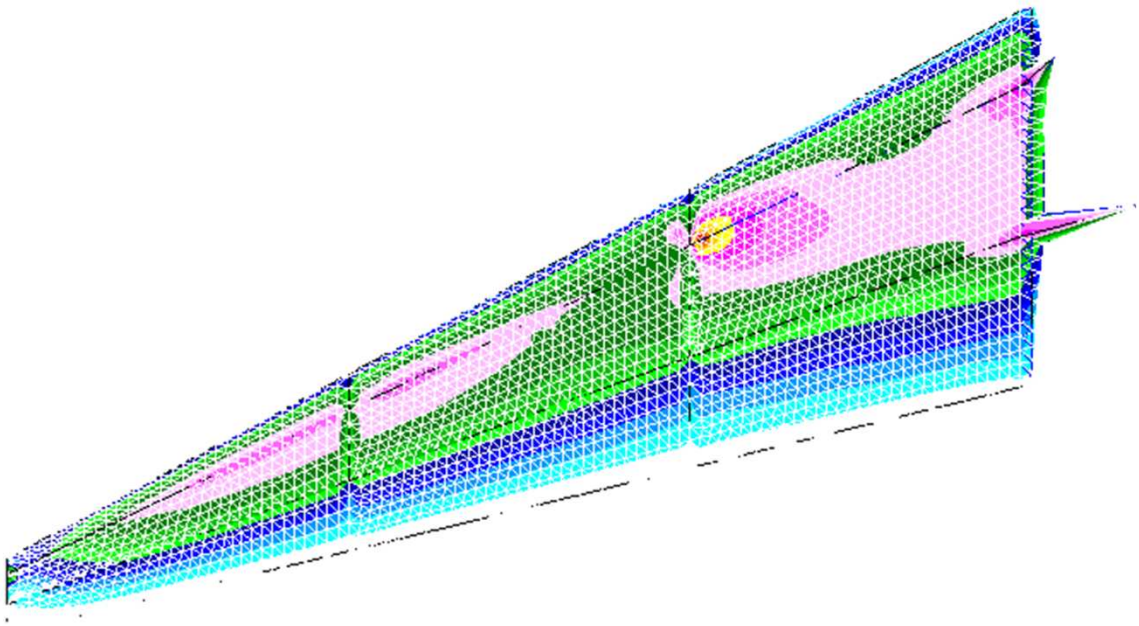
- Create
- Quick Plot
- Select Fringe Result
- Select Deformation Result
- Apply

9.1 Analyze Model in Nastran and Results: SOL101 Linear Static

If you select Displacement, Translational → Displacement, Translational

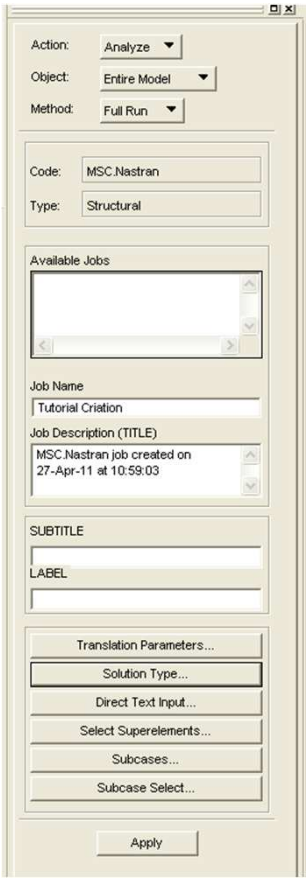


If you select Stress Tensor→ Constraint Forces, Translational



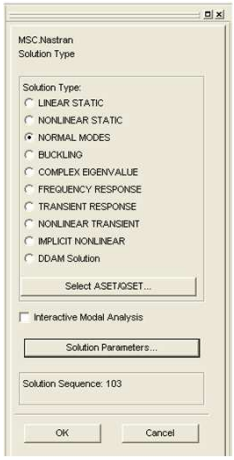
9.2 Analyze Model in Nastran and Results: SOL103 Normal Modes

After we'll try now a NORMAL MODES Solution (SOL103).



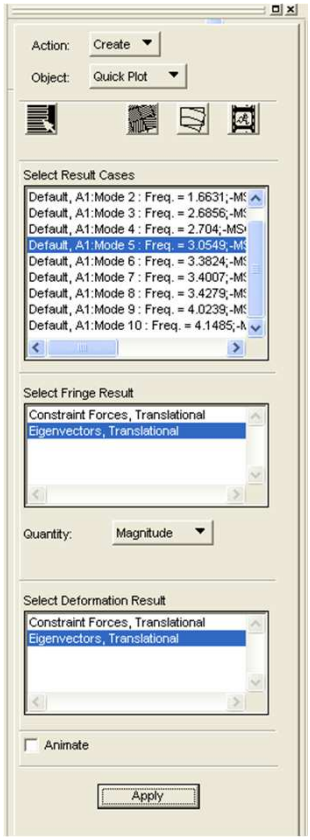
- Analyze
- Entire Model
- Full Run

- Go to Solution Type…
- Select NORMAL MODES (SOL103)
- Ok
- Apply



- Access Results
- Attach XDB
- Results Entities
- Select Result File

- Select the correct XDB file
- Apply

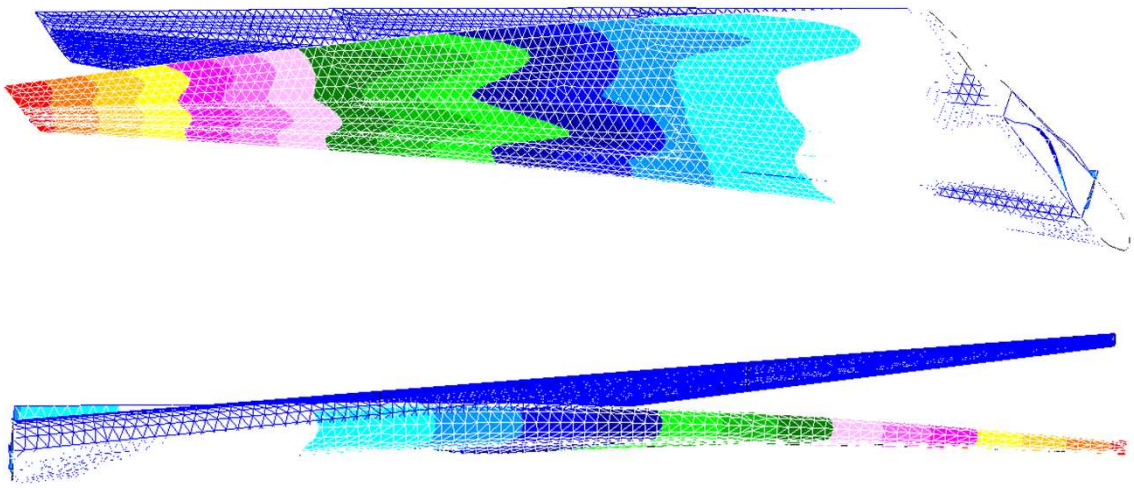


To visualize the results from Nastran analysis go again to *Results* in Patran's toolbox. It's the 2nd box from right to left.

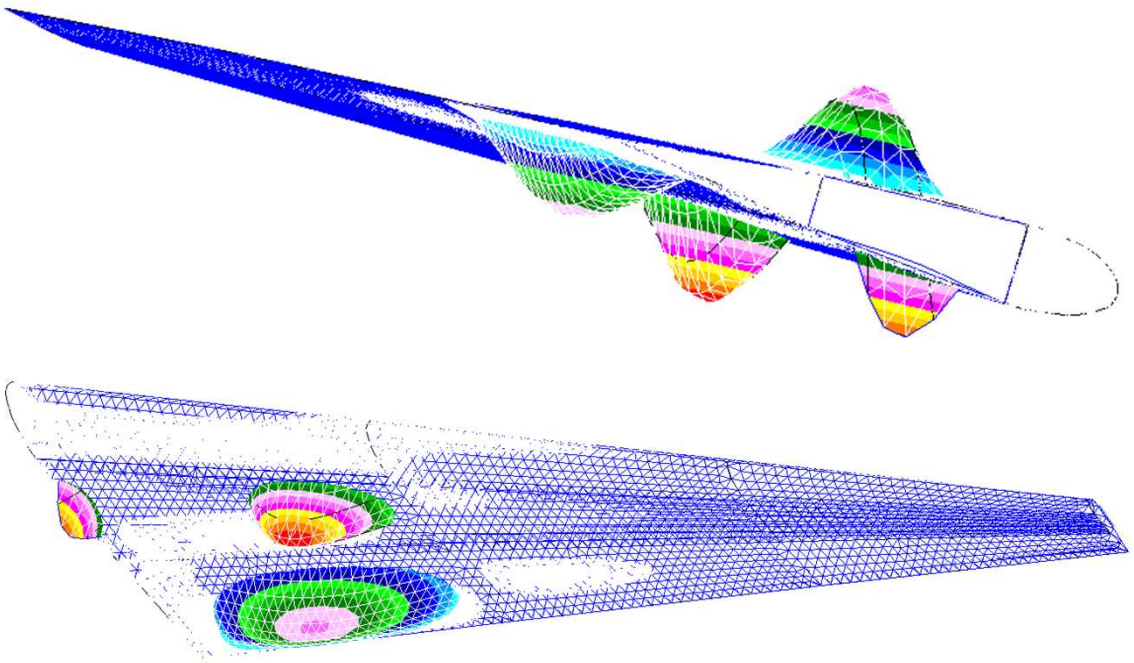
- Create
- Quick Plot
- Select Fringe Result
- Select Deformation Result
- Apply

9.2 Analyze Model in Nastran and Results: SOL103 Normal Modes

If you select Normal Mode 5 → Eigenvectors, Translational → Eigenvectors, Translational



If you select Normal Mode 9 → Eigenvectors, Translational → Eigenvectors, Translational



10.1 Annex PCL Code

```
uil_pref_analysis.set_analysis_preference( "MD Nastran", "Structural", ".bdf", @
 ".op2", "Legacy Mapping" )

$# ----------------------------------------Declare variables----------------------------------------

REAL Span = 30
REAL AR = 9
REAL TRatio = 0.2
REAL Sweep25 = 25
REAL Dihedral = 6
REAL Torsion = -3

REAL Ealum = 69e9
REAL nialum = 0.3
REAL rhoalum = 2770

REAL Esteel = 200e9
REAL nisteel = 0.3
REAL rhosteel = 7800

REAL Etitanium = 120e9
REAL nititanium = 0.3
REAL rhotitanium = 4110

REAL HL = 0.03
REAL WL = 0.03
REAL t1L = 0.002
REAL t2L = 0.002
REAL offL = 0.01

REAL Hhat = 0.007
REAL that = 0.0008
REAL What = 0.01
REAL W1hat = 0.003
REAL offhat = 0.009708

REAL skinthick = 0.002
REAL WBskinthick = 0.002
REAL ribsthick = 0.005
REAL sparsthick = 0.01

$# ----------------------------------------Parametric variables----------------------------------------

REAL nRibs = 4
REAL Semispan = `Span/2`
REAL Cr = `2*Span/AR/(1+TRatio)`
REAL Zm = `Span/2/(nRibs-1)`
REAL pi = 3.1415926535

$# ----------------------------------------------Display----------------------------------------------

ga_viewport_background_set( "default_viewport", 7 )
ga_display_edgecolor_set( "general", 1 )
point_color( 1 )
curve_color( 0 )
```

```
surface_color( 4 )
point_size( 9 )
ga_display_diffuse_set( "general", 1. )

$# ----------------------------------------Points of profile.dat----------------------------------------
STRING asm_create_grid_xyz_created_ids[VIRTUAL]
asm_const_grid_xyz( "1", "[   0.000000  0.017700 0 ] [   0.002300  0.03090" // @
"0 0 ] [   0.005000  0.037200 0 ] [   0.007600  0.041500 0 ] [   0.014300 " // @
" 0.049900 0 ] [   0.024900  0.058200 0 ] [   0.049500  0.073000 0 ] [   0" // @
".074000  0.081400 0 ] [   0.099000  0.086600 0 ] [   0.153000  0.090700 0" // @
" ] [   0.196100  0.090500 0 ] [   0.250400  0.088700 0 ] [   0.309400  0." // @
"085800 0 ] [   0.352000  0.083300 0 ] [   0.391900  0.080400 0 ] [   0.44" // @
"7700  0.075600 0 ] [   0.503400  0.069600 0 ] [   0.559300  0.062600 0 ] " // @
"[   0.596500  0.057500 0 ] [   0.648800  0.049800 0 ] [   0.835100  0.022" // @
"400 0 ] [   0.910900  0.013200 0 ] [   1.000000  0.000300 0 ] [   0.00000" // @
"0  0.017700 0 ] [   0.002200  0.003800 0 ] [   0.004900 -0.001800 0 ] [  " // @
" 0.007200 -0.005300 0 ] [   0.011900 -0.010600 0 ] [   0.024300 -0.020400" // @
" 0 ] [   0.048600 -0.034200 0 ] [   0.071600 -0.045700 0 ] [   0.097900 -" // @
"0.051600 0 ] [   0.148800 -0.060700 0 ] [   0.195300 -0.063200 0 ] [   0." // @
"250100 -0.063200 0 ] [   0.294500 -0.062600 0 ] [   0.357900 -0.061000 0 " // @
"] [   0.396500 -0.059500 0 ] [   0.454300 -0.056300 0 ] [   0.505000 -0.0" // @
"52700 0 ] [   0.555600 -0.048200 0 ] [   0.606300 -0.042700 0 ] [   0.648" // @
"500 -0.037500 0 ] [   0.831700 -0.014900 0 ] [   0.941000 -0.005300 0 ] [" // @
"   1.000000 -0.000300 0 ]", "Coord 0", asm_create_grid_xyz_created_ids )
$? YESFORALL 1000034

$# ----------------------------------Delete duplicated points-------------------------------------------

STRING asm_delete_any_deleted_ids[VIRTUAL]
asm_delete_point( "Point 46 24", asm_delete_any_deleted_ids )

$# ----------------------------------------B-Spline the profile-----------------------------------------

STRING sgm_curve_bspline_created_ids[VIRTUAL]
sgm_const_curve_bspline( "1", "Point 1:9", 5, TRUE, 1, FALSE,  @
sgm_curve_bspline_created_ids )
sgm_const_curve_bspline( "2", "Point 9:17", 5, TRUE, 1, FALSE,  @
sgm_curve_bspline_created_ids )
sgm_const_curve_bspline( "3", "Point 17:23", 4, TRUE, 1, FALSE,  @
sgm_curve_bspline_created_ids )
sgm_const_curve_bspline( "4", "Point 1 25:32", 2, TRUE, 1, FALSE,  @
sgm_curve_bspline_created_ids )
sgm_const_curve_bspline( "5", "Point 32:39", 2, TRUE, 1, FALSE,  @
sgm_curve_bspline_created_ids )
sgm_const_curve_bspline( "6", "Point 39:45 23", 2, TRUE, 1, FALSE,  @
sgm_curve_bspline_created_ids )

$# ------------------------------------Delete points but LE and TE-----------------------------------

STRING asm_delete_any_deleted_ids[VIRTUAL]
asm_delete_point( "Point 2:22 25:45", asm_delete_any_deleted_ids )

$# ----------------------------Merge Upper and Lower camber curves-----------------------------

STRING sgm_edit_curve_merg_created_ids[VIRTUAL]
sgm_edit_curve_merge( "7", "Curve 1:3", 1, 4.9999999E-005, TRUE,  @
sgm_edit_curve_merg_created_ids )
```

```
$? YES 38000217
sgm_edit_curve_merge( "8", "Curve 4:6", 1, 4.9999999E-005, TRUE,  @
sgm_edit_curve_merg_created_ids )
$? YES 38000217

$# --------------------------------Criate vertical curves for the Spars-----------------------------

STRING asm_create_line_xyz_created_ids[VIRTUAL]
asm_const_line_xyz( "9", "<0 2 0>", "[0.2 -1 0] [0.6 -1 0]", "Coord 0",  @
asm_create_line_xyz_created_ids )

$# --------------------------------Intersect vertical curves with profile-----------------------------

STRING asm_create_grid_int_created_ids[VIRTUAL]
asm_const_grid_intersect_v1( "28", "Curve 9 10", "Curve 7 8",  @
asm_create_grid_int_created_ids )

$# ---------------------------------------Delete vertical curves-----------------------------------------

STRING asm_delete_any_deleted_ids[VIRTUAL]
asm_delete_curve( "Curve 9 10", asm_delete_any_deleted_ids )

$# --------------------------------Break the profile curves by the points-------------------------

STRING sgm_curve_break_poi_created_ids[VIRTUAL]
sgm_edit_curve_break_point( "9", "Point 28:31", "Curve 7 8", TRUE,  @
sgm_curve_break_poi_created_ids )
$? YES 38000217

$# ----------------------------------------Verify and Fix-------------------------------------------------

STRING asm_delete_any_deleted_ids[VIRTUAL]
asm_delete_point( "Point 32", asm_delete_any_deleted_ids )
STRING sgm_curve_break_poi_created_ids[VIRTUAL]
sgm_edit_curve_break_point( "15", "Point 29", "Curve 10", TRUE,  @
sgm_curve_break_poi_created_ids )
$? YES 38000217
STRING sgm_edit_curve_merg_created_ids[VIRTUAL]
sgm_edit_curve_merge( "17", "Curve 13 14", 1, 4.9999999E-005, TRUE,  @
sgm_edit_curve_merg_created_ids )
$? YES 38000217

$# --------------------------------Create Wingbox limit curves---------------------------------

STRING asm_line_2point_created_ids[VIRTUAL]
asm_const_line_2point( "18", "Point 28", "Point 29", 0, "", 50., 1,  @
asm_line_2point_created_ids )
asm_const_line_2point( "19", "Point 30", "Point 31", 0, "", 50., 1,  @
asm_line_2point_created_ids )
asm_const_line_2point( "20", "Point 28", "Point 30", 0, "", 50., 1,  @
asm_line_2point_created_ids )
asm_const_line_2point( "21", "Point 29", "Point 31", 0, "", 50., 1,  @
asm_line_2point_created_ids )

$# --------------------------------Criate vertical curves for the Strips-----------------------------

STRING asm_create_line_xyz_created_ids[VIRTUAL]
```

```
asm_const_line_xyz( "22", "<0 2 0>", "[0.333333 -1 0] [0.466667 -1 0]",  @
"Coord 0", asm_create_line_xyz_created_ids )

$# --------------------------------Intersect vertical curves with profile----------------------------

STRING asm_create_grid_int_created_ids[VIRTUAL]
asm_const_grid_intersect_v1( "36", "Curve 22 23", "Curve 18 19",  @
asm_create_grid_int_created_ids )

$# ---------------------------------------Delete vertical curves-----------------------------------------

STRING asm_delete_any_deleted_ids[VIRTUAL]
asm_delete_curve( "Curve 22 23", asm_delete_any_deleted_ids )

$# --------------------------------Break the profile curves by the points-------------------------

STRING sgm_curve_break_poi_created_ids[VIRTUAL]
sgm_edit_curve_break_point( "22", "Point 36:39", "Curve 18 19", TRUE,  @
sgm_curve_break_poi_created_ids )
$? YES 38000217

$# ----------------------------------------Verify and Fix-----------------------------------------------

asm_delete_point( "Point 40", asm_delete_any_deleted_ids )
sgm_edit_curve_break_point( "28", "Point 37", "Curve 23", TRUE,  @
sgm_curve_break_poi_created_ids )
$? YES 38000217
STRING sgm_edit_curve_merg_created_ids[VIRTUAL]
sgm_edit_curve_merge( "30", "Curve 26 27", 1, 0.00050000002, TRUE,  @
sgm_edit_curve_merg_created_ids )
$? YES 38000217

$# --------------------------------Scale the nondimensional profile-------------------------------

STRING sgm_transform_curve_created_ids[VIRTUAL]
sgm_transform_scale( "31", "curve", [ `Cr` `Cr` `Cr` ], "[0 0 0]", "Coord 0",  @
1, TRUE, "Curve 9 11 12 15:17 20:22 24 25 28:30",  @
sgm_transform_curve_created_ids )
$? YES 38000217

$# ---------------------------------------Delete all Points-----------------------------------------

STRING asm_delete_any_deleted_ids[VIRTUAL]
asm_delete_point( "Point 1 23 28:31 36:49", asm_delete_any_deleted_ids )

$# ---------------------------------------Renumber all curves---------------------------------------

STRING sgm_renum_curve_new_ids[VIRTUAL]
sgm_renumber( 1, "curve", "1", "Curve 31", sgm_renum_curve_new_ids )
repaint_graphics(  )
sgm_renumber( 1, "curve", "2", "Curve 39", sgm_renum_curve_new_ids )
repaint_graphics(  )
sgm_renumber( 1, "curve", "3", "Curve 42", sgm_renum_curve_new_ids )
repaint_graphics(  )
sgm_renumber( 1, "curve", "4", "Curve 43", sgm_renum_curve_new_ids )
repaint_graphics(  )
sgm_renumber( 1, "curve", "5", "Curve 35", sgm_renum_curve_new_ids )
```

```
repaint_graphics(  )
sgm_renumber( 1, "curve", "6", "Curve 32", sgm_renum_curve_new_ids )
repaint_graphics(  )
sgm_renumber( 1, "curve", "7", "Curve 40", sgm_renum_curve_new_ids )
repaint_graphics(  )
sgm_renumber( 1, "curve", "8", "Curve 41", sgm_renum_curve_new_ids )
repaint_graphics(  )
sgm_renumber( 1, "curve", "9", "Curve 44", sgm_renum_curve_new_ids )
repaint_graphics(  )
sgm_renumber( 1, "curve", "10", "Curve 36", sgm_renum_curve_new_ids )
repaint_graphics(  )
sgm_renumber( 1, "curve", "11", "Curve 34", sgm_renum_curve_new_ids )
repaint_graphics(  )
sgm_renumber( 1, "curve", "12", "Curve 33", sgm_renum_curve_new_ids )
repaint_graphics(  )
sgm_renumber( 1, "curve", "13", "Curve 37", sgm_renum_curve_new_ids )
repaint_graphics(  )
sgm_renumber( 1, "curve", "14", "Curve 38", sgm_renum_curve_new_ids )
repaint_graphics(  )


$# ----------------------------------------Wing Creation---------------------------------------------------

$# ----------------------------------------Create the other Ribs-------------------------------------

Real i1 = 1
Real i2 = 2
Real i3 = 3
Real i4 = 4

$# ----------------------------------------------RIB 2-----------------------------------------------------
$# ----------------------------------------------Scale-----------------------------------------------------

STRING sgm_transform_curve_created_ids[VIRTUAL]
sgm_transform_scale( "15", "curve", [ `1-(1-TRatio)*(i2-1)/(nRibs-1)` `1-( @
1-TRatio)*(i2-1)/(nRibs-1)` `1-(1-TRatio)*(i2-1)/(nRibs-1)` ], "[0 0 0]",  @
"Coord 0", 1, FALSE, "Curve 1:14", sgm_transform_curve_created_ids )

$# ----------------------------------------------Translate-------------------------------------------------

STRING sgm_transform_curve_created_ids[VIRTUAL]
sgm_transform_translate_v1( "29", "curve",  @
"<`(i2-1)*Zm*(mth_tanr(Sweep25*pi/180)+(1-TRatio)/(AR*(1+TRatio)))` `(i2-1)" // @
"*Zm*mth_tand(Dihedral)` `(i2-1)*Zm`>", `mth_sqrt(((i2-1)*Zm*(mth_tanr( @
Sweep25*pi/180)+(1-TRatio)/(AR*(1+TRatio))))*((i2-1)*Zm*(mth_tanr( @
Sweep25*pi/180)+(1-TRatio)/(AR*(1+TRatio))))+((i2-1)*Zm*mth_tand(Dihedral))*(( @
i2-1)*Zm*mth_tand(Dihedral))+((i2-1)*Zm)*((i2-1)*Zm))`, FALSE, "Coord 0", 1,  @
TRUE, "Curve 15:28", sgm_transform_curve_created_ids )
$? YES 38000217

$# ----------------------------------------------Rotate----------------------------------------------------

STRING sgm_transform_curve_created_ids[VIRTUAL]
sgm_transform_rotate( "43", "curve", "{[`(i2-1)*Zm*(mth_tanr(Sweep25*pi/1" // @
"80)+(1-TRatio)/(AR*(1+TRatio)))` `(i2-1)*Zm*mth_tand(Dihedral)` `(i2-1)*Zm`] " // @
"[0 0 1]}", `(i2-1)*Zm/(Semispan)*Torsion`, 0., "Coord 0", 1, TRUE,  @
"Curve 29:42", sgm_transform_curve_created_ids )
```

```
$? YES 38000217
repaint_graphics(  )

$# ----------------------------------------------RIB 3----------------------------------------------
$# ----------------------------------------------Scale----------------------------------------------
STRING sgm_transform_curve_created_ids[VIRTUAL]
sgm_transform_scale( "57", "curve", [ `1-(1-TRatio)*(i3-1)/(nRibs-1)` `1-( @
1-TRatio)*(i3-1)/(nRibs-1)` `1-(1-TRatio)*(i3-1)/(nRibs-1)` ], "[0 0 0]",  @
"Coord 0", 1, FALSE, "Curve 1:14", sgm_transform_curve_created_ids )

$# ----------------------------------------------Translate----------------------------------------------

STRING sgm_transform_curve_created_ids[VIRTUAL]
sgm_transform_translate_v1( "71", "curve",  @
"<`(i3-1)*Zm*(mth_tanr(Sweep25*pi/180)+(1-TRatio)/(AR*(1+TRatio)))` `(i3-1)" // @
"*Zm*mth_tand(Dihedral)` `(i3-1)*Zm`>", `mth_sqrt(((i3-1)*Zm*(mth_tanr( @
Sweep25*pi/180)+(1-TRatio)/(AR*(1+TRatio))))*((i3-1)*Zm*(mth_tanr( @
Sweep25*pi/180)+(1-TRatio)/(AR*(1+TRatio))))+((i3-1)*Zm*mth_tand(Dihedral))*(( @
i3-1)*Zm*mth_tand(Dihedral))+((i3-1)*Zm)*((i3-1)*Zm))`, FALSE, "Coord 0", 1,  @
TRUE, "Curve 57:70", sgm_transform_curve_created_ids )
$? YES 38000217

$# ----------------------------------------------Rotate----------------------------------------------

STRING sgm_transform_curve_created_ids[VIRTUAL]
sgm_transform_rotate( "85", "curve", "{[`(i3-1)*Zm*(mth_tanr(Sweep25*pi/1" // @
"80)+(1-TRatio)/(AR*(1+TRatio)))` `(i3-1)*Zm*mth_tand(Dihedral)` `(i3-1)*Zm`] " // @
"[0 0 1]}", `(i3-1)*Zm/(Semispan)*Torsion`, 0., "Coord 0", 1, TRUE,  @
"Curve 71:84", sgm_transform_curve_created_ids )
$? YES 38000217
repaint_graphics(  )

$# ----------------------------------------------RIB 4----------------------------------------------
$# ----------------------------------------------Scale----------------------------------------------
STRING sgm_transform_curve_created_ids[VIRTUAL]
sgm_transform_scale( "99", "curve", [ `1-(1-TRatio)*(i4-1)/(nRibs-1)` `1-( @
1-TRatio)*(i4-1)/(nRibs-1)` `1-(1-TRatio)*(i4-1)/(nRibs-1)` ], "[0 0 0]",  @
"Coord 0", 1, FALSE, "Curve 1:14", sgm_transform_curve_created_ids )

$# ----------------------------------------------Translate----------------------------------------------

STRING sgm_transform_curve_created_ids[VIRTUAL]
sgm_transform_translate_v1( "113", "curve",  @
"<`(i4-1)*Zm*(mth_tanr(Sweep25*pi/180)+(1-TRatio)/(AR*(1+TRatio)))` `(i4-1)" // @
"*Zm*mth_tand(Dihedral)` `(i4-1)*Zm`>", `mth_sqrt(((i4-1)*Zm*(mth_tanr( @
Sweep25*pi/180)+(1-TRatio)/(AR*(1+TRatio))))*((i4-1)*Zm*(mth_tanr( @
Sweep25*pi/180)+(1-TRatio)/(AR*(1+TRatio))))+((i4-1)*Zm*mth_tand(Dihedral))*(( @
i4-1)*Zm*mth_tand(Dihedral))+((i4-1)*Zm)*((i4-1)*Zm))`, FALSE, "Coord 0", 1,  @
TRUE, "Curve 99:112", sgm_transform_curve_created_ids )
$? YES 38000217

$# ----------------------------------------------Rotate----------------------------------------------

STRING sgm_transform_curve_created_ids[VIRTUAL]
sgm_transform_rotate( "127", "curve", "{[`(i4-1)*Zm*(mth_tanr(Sweep25*pi/1" // @
"80)+(1-TRatio)/(AR*(1+TRatio)))` `(i4-1)*Zm*mth_tand(Dihedral)` `(i4-1)*Zm`] " // @
"[0 0 1]}", `(i4-1)*Zm/(Semispan)*Torsion`, 0., "Coord 0", 1, TRUE,  @
```

```
"Curve 113:126", sgm_transform_curve_created_ids )
$? YES 38000217
repaint_graphics(  )

$# ----------------------------------------------Delete all points-----------------------------------------

STRING asm_delete_any_deleted_ids[VIRTUAL]
asm_delete_point( "Point 11 21:29 40 50:58 69 79:87",  @
asm_delete_any_deleted_ids )
repaint_graphics(  )

$# ----------------------------------------------Create the Srip curves-------------------------------------

STRING asm_line_2point_created_ids[VIRTUAL]
asm_const_line_2point( " 141 ", "Curve  1.1 ", "Curve  43.1 ", 0, "", 50., 1,  @
asm_line_2point_created_ids )
asm_const_line_2point( " 142 ", "Curve  43.1 ", "Curve  85.1 ", 0, "", 50., 1,  @
asm_line_2point_created_ids )
asm_const_line_2point( " 143 ", "Curve  85.1 ", "Curve  127.1 ", 0, "", 50., 1,  @
asm_line_2point_created_ids )
asm_const_line_2point( " 144 ", "Curve  11.1 ", "Curve  53.1 ", 0, "", 50., 1,  @
asm_line_2point_created_ids )
asm_const_line_2point( " 145 ", "Curve  53.1 ", "Curve  95.1 ", 0, "", 50., 1,  @
asm_line_2point_created_ids )
asm_const_line_2point( " 146 ", "Curve  95.1 ", "Curve  137.1 ", 0, "", 50., 1,  @
asm_line_2point_created_ids )
asm_const_line_2point( " 147 ", "Curve  3.1 ", "Curve  45.1 ", 0, "", 50., 1,  @
asm_line_2point_created_ids )
asm_const_line_2point( " 148 ", "Curve  45.1 ", "Curve  87.1 ", 0, "", 50., 1,  @
asm_line_2point_created_ids )
asm_const_line_2point( " 149 ", "Curve  87.1 ", "Curve  129.1 ", 0, "", 50., 1,  @
asm_line_2point_created_ids )
asm_const_line_2point( " 150 ", "Curve  4.1 ", "Curve  46.1 ", 0, "", 50., 1,  @
asm_line_2point_created_ids )
asm_const_line_2point( " 151 ", "Curve  46.1 ", "Curve  88.1 ", 0, "", 50., 1,  @
asm_line_2point_created_ids )
asm_const_line_2point( " 152 ", "Curve  88.1 ", "Curve  130.1 ", 0, "", 50., 1,  @
asm_line_2point_created_ids )
asm_const_line_2point( " 153 ", "Curve  5.1 ", "Curve  47.1 ", 0, "", 50., 1,  @
asm_line_2point_created_ids )
asm_const_line_2point( " 154 ", "Curve  47.1 ", "Curve  89.1 ", 0, "", 50., 1,  @
asm_line_2point_created_ids )
asm_const_line_2point( " 155 ", "Curve  89.1 ", "Curve  131.1 ", 0, "", 50., 1,  @
asm_line_2point_created_ids )
asm_const_line_2point( " 156 ", "Curve  10.2 ", "Curve  52.2 ", 0, "", 50., 1,  @
asm_line_2point_created_ids )
asm_const_line_2point( " 157 ", "Curve  52.2 ", "Curve  94.2 ", 0, "", 50., 1,  @
asm_line_2point_created_ids )
asm_const_line_2point( " 158 ", "Curve  94.2 ", "Curve  136.2 ", 0, "", 50., 1,  @
asm_line_2point_created_ids )
asm_const_line_2point( " 159 ", "Curve  12.1 ", "Curve  54.1 ", 0, "", 50., 1,  @
asm_line_2point_created_ids )
asm_const_line_2point( " 160 ", "Curve  54.1 ", "Curve  96.1 ", 0, "", 50., 1,  @
asm_line_2point_created_ids )
asm_const_line_2point( " 161 ", "Curve  96.1 ", "Curve  138.1 ", 0, "", 50., 1,  @
asm_line_2point_created_ids )
asm_const_line_2point( " 162 ", "Curve  8.1 ", "Curve  50.1 ", 0, "", 50., 1,  @
```

```
asm_line_2point_created_ids )
asm_const_line_2point( " 163 ", "Curve  50.1 ", "Curve  92.1 ", 0, "", 50., 1,  @
asm_line_2point_created_ids )
asm_const_line_2point( " 164 ", "Curve  92.1 ", "Curve  134.1 ", 0, "", 50., 1,  @
asm_line_2point_created_ids )
asm_const_line_2point( " 165 ", "Curve  9.1 ", "Curve  51.1 ", 0, "", 50., 1,  @
asm_line_2point_created_ids )
asm_const_line_2point( " 166 ", "Curve  51.1 ", "Curve  93.1 ", 0, "", 50., 1,  @
asm_line_2point_created_ids )
asm_const_line_2point( " 167 ", "Curve  93.1 ", "Curve  135.1 ", 0, "", 50., 1,  @
asm_line_2point_created_ids )
asm_const_line_2point( " 168 ", "Curve  10.1 ", "Curve  52.1 ", 0, "", 50., 1,  @
asm_line_2point_created_ids )
asm_const_line_2point( " 169 ", "Curve  52.1 ", "Curve  94.1 ", 0, "", 50., 1,  @
asm_line_2point_created_ids )
asm_const_line_2point( " 170 ", "Curve  94.1 ", "Curve  136.1 ", 0, "", 50., 1,  @
asm_line_2point_created_ids )


$# ----------------------------------------------Create Ribs surfaces-------------------------------------

STRING sgm_surface_trimmed__created_id[VIRTUAL]
sgm_create_surface_trimmed_v1( "1", "Curve 1 11 5 10 12 6", "", "",  @
FALSE, TRUE, TRUE, TRUE, sgm_surface_trimmed__created_id )
sgm_create_surface_trimmed_v1( "2", "Curve 43 53 47 52 54 48", "", "",  @
FALSE, TRUE, TRUE, TRUE, sgm_surface_trimmed__created_id )
sgm_create_surface_trimmed_v1( "3", "Curve 85 95 89 94 96 90", "", "",  @
FALSE, TRUE, TRUE, TRUE, sgm_surface_trimmed__created_id )
sgm_create_surface_trimmed_v1( "4", "Curve 127 137 131 136 138 132", "", "",  @
FALSE, TRUE, TRUE, TRUE, sgm_surface_trimmed__created_id )

$# ----------------------------------------------Create Spars surfaces-------------------------------------

STRING sgm_surface_trimmed__created_id[VIRTUAL]
sgm_create_surface_trimmed_v1( "5", "Curve 13 144:146 139 161:159:-1", "", "", @
 FALSE, TRUE, TRUE, TRUE, sgm_surface_trimmed__created_id )
sgm_create_surface_trimmed_v1( "6", "Curve 14 153:155 140 170:168:-1", "", "", @
 FALSE, TRUE, TRUE, TRUE, sgm_surface_trimmed__created_id )

$# ----------------------------------------------Create WB skin surfaces-------------------------------------

STRING sgm_surface_4edge_created_ids[VIRTUAL]
sgm_const_surface_4edge( " 7 ", "Curve  144 ", "Curve  2 ", "Curve  147 ",  @
"Curve  44 ", sgm_surface_4edge_created_ids )
sgm_const_surface_4edge( " 8 ", "Curve  145 ", "Curve  44 ", "Curve  148 ",  @
"Curve  86 ", sgm_surface_4edge_created_ids )
sgm_const_surface_4edge( " 9 ", "Curve  146 ", "Curve  86 ", "Curve  149 ",  @
"Curve  128 ", sgm_surface_4edge_created_ids )
sgm_const_surface_4edge( " 10 ", "Curve  147 ", "Curve  3 ", "Curve  150 ",  @
"Curve  45 ", sgm_surface_4edge_created_ids )
sgm_const_surface_4edge( " 11 ", "Curve  148 ", "Curve  45 ", "Curve  151 ",  @
"Curve  87 ", sgm_surface_4edge_created_ids )
sgm_const_surface_4edge( " 12 ", "Curve  149 ", "Curve  87 ", "Curve  152 ",  @
"Curve  129 ", sgm_surface_4edge_created_ids )
sgm_const_surface_4edge( " 13 ", "Curve  150 ", "Curve  4 ", "Curve  153 ",  @
"Curve  46 ", sgm_surface_4edge_created_ids )
sgm_const_surface_4edge( " 14 ", "Curve  151 ", "Curve  46 ", "Curve  154 ",  @
```

```
"Curve  88 ", sgm_surface_4edge_created_ids )
sgm_const_surface_4edge( " 15 ", "Curve  152 ", "Curve  88 ", "Curve  155 ",  @
"Curve  130 ", sgm_surface_4edge_created_ids )
sgm_const_surface_4edge( " 16 ", "Curve  159 ", "Curve  7 ", "Curve  162 ",  @
"Curve  49 ", sgm_surface_4edge_created_ids )
sgm_const_surface_4edge( " 17 ", "Curve  160 ", "Curve  49 ", "Curve  163 ",  @
"Curve  91 ", sgm_surface_4edge_created_ids )
sgm_const_surface_4edge( " 18 ", "Curve  161 ", "Curve  91 ", "Curve  164 ",  @
"Curve  133 ", sgm_surface_4edge_created_ids )
sgm_const_surface_4edge( " 19 ", "Curve  162 ", "Curve  8 ", "Curve  165 ",  @
"Curve  50 ", sgm_surface_4edge_created_ids )
sgm_const_surface_4edge( " 20 ", "Curve  163 ", "Curve  50 ", "Curve  166 ",  @
"Curve  92 ", sgm_surface_4edge_created_ids )
sgm_const_surface_4edge( " 21 ", "Curve  164 ", "Curve  92 ", "Curve  167 ",  @
"Curve  134 ", sgm_surface_4edge_created_ids )
sgm_const_surface_4edge( " 22 ", "Curve  165 ", "Curve  9 ", "Curve  168 ",  @
"Curve  51 ", sgm_surface_4edge_created_ids )
sgm_const_surface_4edge( " 23 ", "Curve  166 ", "Curve  51 ", "Curve  169 ",  @
"Curve  93 ", sgm_surface_4edge_created_ids )
sgm_const_surface_4edge( " 24 ", "Curve  167 ", "Curve  93 ", "Curve  170 ",  @
"Curve  135 ", sgm_surface_4edge_created_ids )


$# ----------------------------------Create Upper Camber skin surfaces-------------------------------

sgm_const_surface_4edge( " 25 ", "Curve  1 ", "Curve  141 ", "Curve  43 ",  @
"Curve  144 ", sgm_surface_4edge_created_ids )
sgm_const_surface_4edge( " 26 ", "Curve  43 ", "Curve  142 ", "Curve  85 ",  @
"Curve  145 ", sgm_surface_4edge_created_ids )
sgm_const_surface_4edge( " 27 ", "Curve  85 ", "Curve  143 ", "Curve  127 ",  @
"Curve  146 ", sgm_surface_4edge_created_ids )
sgm_const_surface_4edge( " 28 ", "Curve  11 ", "Curve  144 ", "Curve  53 ",  @
"Curve  153 ", sgm_surface_4edge_created_ids )
sgm_const_surface_4edge( " 29 ", "Curve  53 ", "Curve  145 ", "Curve  95 ",  @
"Curve  154 ", sgm_surface_4edge_created_ids )
sgm_const_surface_4edge( " 30 ", "Curve  95 ", "Curve  146 ", "Curve  137 ",  @
"Curve  155 ", sgm_surface_4edge_created_ids )
sgm_const_surface_4edge( " 31 ", "Curve  5 ", "Curve  153 ", "Curve  47 ",  @
"Curve  156 ", sgm_surface_4edge_created_ids )
sgm_const_surface_4edge( " 32 ", "Curve  47 ", "Curve  154 ", "Curve  89 ",  @
"Curve  157 ", sgm_surface_4edge_created_ids )
sgm_const_surface_4edge( " 33 ", "Curve  89 ", "Curve  155 ", "Curve  131 ",  @
"Curve  158 ", sgm_surface_4edge_created_ids )

$# ----------------------------------Create Lower Camber skin surfaces--------------------------------

sgm_const_surface_4edge( " 34 ", "Curve  6 ", "Curve  141 ", "Curve  48 ",  @
"Curve  159 ", sgm_surface_4edge_created_ids )
sgm_const_surface_4edge( " 35 ", "Curve  48 ", "Curve  142 ", "Curve  90 ",  @
"Curve  160 ", sgm_surface_4edge_created_ids )
sgm_const_surface_4edge( " 36 ", "Curve  90 ", "Curve  143 ", "Curve  132 ",  @
"Curve  161 ", sgm_surface_4edge_created_ids )
sgm_const_surface_4edge( " 37 ", "Curve  12 ", "Curve  159 ", "Curve  54 ",  @
"Curve  168 ", sgm_surface_4edge_created_ids )
sgm_const_surface_4edge( " 38 ", "Curve  54 ", "Curve  160 ", "Curve  96 ",  @
"Curve  169 ", sgm_surface_4edge_created_ids )
sgm_const_surface_4edge( " 39 ", "Curve  96 ", "Curve  161 ", "Curve  138 ",  @
```

```
"Curve  170 ", sgm_surface_4edge_created_ids )
sgm_const_surface_4edge( " 40 ", "Curve  10 ", "Curve  168 ", "Curve  52 ",  @
"Curve  156 ", sgm_surface_4edge_created_ids )
sgm_const_surface_4edge( " 41 ", "Curve  52 ", "Curve  169 ", "Curve  94 ",  @
"Curve  157 ", sgm_surface_4edge_created_ids )
sgm_const_surface_4edge( " 42 ", "Curve  94 ", "Curve  170 ", "Curve  136 ",  @
"Curve  158 ", sgm_surface_4edge_created_ids )

$# ------------------------------------------------Ajust View-----------------------------------------------------

ga_view_aa_set( -168, -10, 165 )
ga_view_zoom_set( 3.5)
repaint_graphics(  )

$# ------------------------------------------------------------------------------------------------------------------
$# ---------------------------------------------Create Materials-----------------------------------------------

$# ---------------------------------------------ALUMINIUM------------------------------------------------------
material.create( "Analysis code ID", 1, "Analysis type ID", 1, "Alum", 0,  @
"Date: 14-Apr-11          Time: 16:25:54", "Isotropic", 1, "Directionality",  @
1, "Linearity", 1, "Homogeneous", 0, "Linear Elastic", 1,  @
"Model Options & IDs", ["", "", "", "", ""], [0, 0, 0, 0, 0], "Active Flag",  @
1, "Create", 10, "External Flag", FALSE, "Property IDs", ["Elastic Modulus",  @
"Poisson Ratio", "Density"], [2, 5, 16, 0], "Property Values", ["`Ealum`", "`nialum`", @
 "`rhoalum`", ""] )

$# ------------------------------------------------STEEL----------------------------------------------------
material.create( "Analysis code ID", 1, "Analysis type ID", 1, "Steel", 0,  @
"Date: 14-Apr-11          Time: 16:25:54", "Isotropic", 1, "Directionality",  @
1, "Linearity", 1, "Homogeneous", 0, "Linear Elastic", 1,  @
"Model Options & IDs", ["", "", "", "", ""], [0, 0, 0, 0, 0], "Active Flag",  @
1, "Create", 10, "External Flag", FALSE, "Property IDs", ["Elastic Modulus",  @
"Poisson Ratio", "Density"], [2, 5, 16, 0], "Property Values", ["`Esteel`", "`nisteel`" @
, "`rhosteel`", ""] )

$# ---------------------------------------------TITANIUM-------------------------------------------------------
material.create( "Analysis code ID", 1, "Analysis type ID", 1, "Titanium", 0,  @
"Date: 14-Apr-11          Time: 16:25:54", "Isotropic", 1, "Directionality",  @
1, "Linearity", 1, "Homogeneous", 0, "Linear Elastic", 1,  @
"Model Options & IDs", ["", "", "", "", ""], [0, 0, 0, 0, 0], "Active Flag",  @
1, "Create", 10, "External Flag", FALSE, "Property IDs", ["Elastic Modulus",  @
"Poisson Ratio", "Density"], [2, 5, 16, 0], "Property Values", ["`Etitanium`", "`nititanium`" @
, "`rhotitanium`", ""] )

$# ------------------------------------------------------------------------------------------------------------------
$# ---------------------------------------------Create Properties----------------------------------------------

$# ---------------------------------------------L-BEAM STRIPS--------------------------------------------------
beam_section_create( "L_Beam_StripsWB", "L", ["`HL`", "`WL`", "`t1L`", "`t2L`" @
] )

$# ---------------------------------------------HAT-BEAM STRIPS------------------------------------------------
beam_section_create( "Hat_Beam_StripsWB", "HAT", ["`Hhat`", "`that`", "`What`",  @
"`W1hat`"] )

$# ---------------------------------------------SHELL SKIN-----------------------------------------------------
elementprops_create( "ShellSkin", 51, 25, 35, 1, 1, 20, [13, 20, 36, 4037, 4111,  @
```

```
4118, 4119], [5, 9, 1, 1, 1, 1, 1], ["m:Alum", "", "`skinthick`", "", "", "", @
 ""], "Surface 25:42" )

$# ------------------------------------SHELL SPARS-----------------------------------------------
elementprops_create( "ShellSpars", 51, 25, 35, 1, 1, 20, [13, 20, 36, 4037,  @
4111, 4118, 4119], [5, 9, 1, 1, 1, 1, 1], ["m:Titanium", "", "`sparsthick`", "",  @
"", ""], "Surface 5 6" )

$# ------------------------------------SHELL SKIN SUP/INF WB-------------------------------------
elementprops_create( "ShellSkinWB", 51, 25, 35, 1, 1, 20, [13, 20, 36, 4037,  @
4111, 4118, 4119], [5, 9, 1, 1, 1, 1, 1], ["m:Alum", "", "`WBskinthick`", "",  @
"", "", ""], "Surface 7:24" )

$# ------------------------------------------SHELL RIBS-----------------------------------------
elementprops_create( "ShellRibs", 51, 25, 35, 1, 1, 20, [13, 20, 36, 4037,  @
4111, 4118, 4119], [5, 9, 1, 1, 1, 1, 1], ["m:Alum", "", "`ribsthick`", "", "", "",  @
""], "Surface 1:4" )

$# ------------------------------------------------Create Hat Strips-----------------------------------------

elementprops_create( "SupWBStrips", 11, 2, 42, 1, 1, 20, [39, 13, 6, 4042,  @
4043, 2047, 2048, 1, 10, 11, 4026, 1026, 4044, 4045, 4037, 4047, 4048, 4050,  @
4051, 4053, 4054, 4056, 4057, 4061, 8200, 8201, 8202], [11, 5, 2, 2, 2, 4, 4,  @
1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 6, 4, 4, 4], [ @
"Hat_Beam_StripsWB", "m:Steel", "<0 -1 0>", "<0 `-offhat` 0>", "<0 `-offhat` 0>", "1", @
 "1", "", "", "", "", "", "", "", "", "", "", "", "", "", "", "", "", "", "", "",  @
"Analysis", "Analysis", "Analysis"], "" )

elementprops_create( "InfWBStrips", 11, 2, 42, 1, 1, 20, [39, 13, 6, 4042,  @
4043, 2047, 2048, 1, 10, 11, 4026, 1026, 4044, 4045, 4037, 4047, 4048, 4050,  @
4051, 4053, 4054, 4056, 4057, 4061, 8200, 8201, 8202], [11, 5, 2, 2, 2, 4, 4,  @
1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 6, 4, 4, 4], [ @
"Hat_Beam_StripsWB", "m:Steel", "<0 1 0>", "<0 `offhat` 0>", "<0 `offhat` 0>", "1",  @
"1", "", "", "", "", "", "", "", "", "", "", "", "", "", "", "", "", "", "", "",  @
"", ""], "" )

$# ------------------------------------------------Create L Strips-----------------------------------------

elementprops_create( "FLStrip", 11, 2, 42, 1, 1, 20, [39, 13, 6, 4042, 4043,  @
2047, 2048, 1, 10, 11, 4026, 1026, 4044, 4045, 4037, 4047, 4048, 4050, 4051,  @
4053, 4054, 4056, 4057, 4061, 8200, 8201, 8202], [11, 5, 2, 2, 2, 4, 4, 1, 1,  @
1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 6, 4, 4, 4], ["L_Beam_StripsWB",  @
"m:Steel", "<1 0 0>", "<`offL` `offL` 0>", "<`offL` `offL` 0>", "1", "1", "", "", "", "", "", "",  @
"", "", "", "", "", "", "", "", "", "", "", "", "", "", "", "", "", "", "", ""], "" )

elementprops_create( "FUStrip", 11, 2, 42, 1, 1, 20, [39, 13, 6, 4042, 4043,  @
2047, 2048, 1, 10, 11, 4026, 1026, 4044, 4045, 4037, 4047, 4048, 4050, 4051,  @
4053, 4054, 4056, 4057, 4061, 8200, 8201, 8202], [11, 5, 2, 2, 2, 4, 4, 1, 1,  @
1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 6, 4, 4, 4], ["L_Beam_StripsWB",  @
"m:Steel", "<0 -1 0>", "<`offL` `-offL` 0>", "<`offL` `-offL` 0>", "1", "1", "", "", "", "", "", "",  @
"", "", "", "", "", "", "", "", "", "", "", "", "", "", "", "", "", "", "", ""], "" )

elementprops_create( "BLStrip", 11, 2, 42, 1, 1, 20, [39, 13, 6, 4042, 4043,  @
2047, 2048, 1, 10, 11, 4026, 1026, 4044, 4045, 4037, 4047, 4048, 4050, 4051,  @
4053, 4054, 4056, 4057, 4061, 8200, 8201, 8202], [11, 5, 2, 2, 2, 4, 4, 1, 1,  @
1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 6, 4, 4, 4], ["L_Beam_StripsWB",  @
"m:Steel", "<0 1 0>", "<`-offL` `offL` 0>", "<`-offL` `offL` 0>", "1", "1", "", "", "", "", "", "",  @
```

```
"", "", "", "", "", "", "", "", "", "", "", "Analysis", "Analysis", "Analysis" @
], "" )

elementprops_create( "BUStrip", 11, 2, 42, 1, 1, 20, [39, 13, 6, 4042, 4043,  @
2047, 2048, 1, 10, 11, 4026, 1026, 4044, 4045, 4037, 4047, 4048, 4050, 4051,  @
4053, 4054, 4056, 4057, 4061, 8200, 8201, 8202], [11, 5, 2, 2, 2, 4, 4, 1, 1,  @
1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 6, 4, 4, 4], ["L_Beam_StripsWB",  @
"m:Steel", "<-1 0 0>", "<`-offL` `-offL` 0>", "<`-offL` `-offL` 0>", "1", "1", "", "", "", "", "", "", @
 "", "", "", "", "", "", "", "", "", "", "", "", "", "", ""], "" )

repaint_graphics(  )

$# ------------------------------------------------MESHING-------------------------------------------------

$# ------------------------------------------------Group Creation------------------------------------------

repaint_graphics(  )
sys_poll_option( 2 )

ga_group_create( "Frame" )
ga_group_entity_add( "Frame", "Curve 1:207" )

ga_group_create( "Ribs" )
ga_group_entity_add( "Ribs", "Surface 1:4 Curve 1 11 5 6 12 10 " // @
"127 137 131 132 138 136 141:143 156:158" )

ga_group_create( "Spars" )
ga_group_entity_add( "Spars", "Surface 5 6 Curve 1 11 5 6 12 10 " // @
"127 137 131 132 138 136 141:143 156:158" )

ga_group_create( "Sup Skin" )
ga_group_entity_add( "Sup Skin", "Surface 25:33 Curve 1 11 5 6 12 10 " // @
"127 137 131 132 138 136 141:143 156:158" )

ga_group_create( "Inf Skin" )
ga_group_entity_add( "Inf Skin", "Surface 34:42 Curve 1 11 5 6 12 10 " // @
"127 137 131 132 138 136 141:143 156:158" )

ga_group_create( "WB Skin" )
ga_group_entity_add( "WB Skin", "Surface 7:24 Curve 1 11 5 6 12 10 " // @
"127 137 131 132 138 136 141:143 156:158" )

ga_group_create( "Exterior" )
ga_group_entity_add( "Exterior", "Surface 1 4 25:42" )

ga_group_create( "MSeed" )
ga_group_entity_add( "MSeed", "Curve 1 11 5 6 12 10 127 137 131 132 138 136 14" // @
"1:143 156:158" )

ga_group_create( "MStrips" )
ga_group_entity_add( "MStrips", "Curve 1 11 5 6 12 10 127 137 131 132 138 136 14" // @
"1:143 156:158" )

ga_group_create( "MSpars" )
ga_group_entity_add( "MSpars", "Curve 1 11 5 6 12 10 127 137 131 132 138 136 14" // @
"1:143 156:158" )
```

```
ga_group_create( "MRibs" )
ga_group_entity_add( "MRibs", "Curve 1 11 5 6 12 10 127 137 131 132 138 136 14" // @
"1:143 156:158" )

ga_group_create( "MWB" )
ga_group_entity_add( "MWB", "Curve 1 11 5 6 12 10 127 137 131 132 138 136 14" // @
"1:143 156:158" )

ga_group_create( "MSkin" )
ga_group_entity_add( "MSkin", "Curve 1 11 5 6 12 10 127 137 131 132 138 136 14" // @
"1:143 156:158" )

$# ----------------------------------------------------Criate Mesh Seed-----------------------------------

sys_poll_option( 0 )
uil_viewport_post_groups.posted_groups( "default_viewport", 1, ["MSeed"] )
repaint_graphics(  )

ui_exec_function( "mesh_seed_display_mgr", "init" )
mesh_seed_create( "Curve 144:155", 1, 31, 0., 0., 0. )
mesh_seed_create( "Curve 159:170", 1, 31, 0., 0., 0. )
mesh_seed_display_mgr.refresh(  )

$# ------------------------------Mesh WB Skin Hat-Strips-----------------------------------------

sys_poll_option( 0 )
uil_viewport_post_groups.posted_groups( "default_viewport", 1, ["MStrips"] )
repaint_graphics(  )

ui_exec_function( "mesh_seed_display_mgr", "init" )
INTEGER fem_create_mesh_curve_num_nodes
INTEGER fem_create_mesh_curve_num_elems
STRING fem_create_mesh_c_nodes_created[VIRTUAL]
STRING fem_create_mesh_c_elems_created[VIRTUAL]
fem_create_mesh_curv_1( "Curve 147:152", 16384, 0.54812503, "Bar2", "#", "#",  @
"Coord 0", "Coord 0", fem_create_mesh_curve_num_nodes,  @
fem_create_mesh_curve_num_elems, fem_create_mesh_c_nodes_created,  @
fem_create_mesh_c_elems_created )
fem_associate_elems_to_ep( "SupWBStrips", "1:186", 186 )

INTEGER fem_create_mesh_curve_num_nodes
INTEGER fem_create_mesh_curve_num_elems
STRING fem_create_mesh_c_nodes_created[VIRTUAL]
STRING fem_create_mesh_c_elems_created[VIRTUAL]
fem_create_mesh_curv_1( "Curve 162:167", 16384, 0.55001199, "Bar2", "#", "#",  @
"Coord 0", "Coord 0", fem_create_mesh_curve_num_nodes,  @
fem_create_mesh_curve_num_elems, fem_create_mesh_c_nodes_created,  @
fem_create_mesh_c_elems_created )
fem_associate_elems_to_ep( "InfWBStrips", "187:372", 186 )


$# ------------------------------Mesh WB corner L-Strips-----------------------------------------

INTEGER fem_create_mesh_curve_num_nodes
INTEGER fem_create_mesh_curve_num_elems
STRING fem_create_mesh_c_nodes_created[VIRTUAL]
STRING fem_create_mesh_c_elems_created[VIRTUAL]
```

```
fem_create_mesh_curv_1( "Curve 144:146", 16384, 0.55628097, "Bar2", "#", "#",  @
"Coord 0", "Coord 0", fem_create_mesh_curve_num_nodes,  @
fem_create_mesh_curve_num_elems, fem_create_mesh_c_nodes_created,  @
fem_create_mesh_c_elems_created )
fem_associate_elems_to_ep( "FUStrip", "373:465", 93 )

INTEGER fem_create_mesh_curve_num_nodes
INTEGER fem_create_mesh_curve_num_elems
STRING fem_create_mesh_c_nodes_created[VIRTUAL]
STRING fem_create_mesh_c_elems_created[VIRTUAL]
fem_create_mesh_curv_1( "Curve 159:161", 16384, 0.55833501, "Bar2", "#", "#",  @
"Coord 0", "Coord 0", fem_create_mesh_curve_num_nodes,  @
fem_create_mesh_curve_num_elems, fem_create_mesh_c_nodes_created,  @
fem_create_mesh_c_elems_created )
fem_associate_elems_to_ep( "FLStrip", "466:558", 93 )

INTEGER fem_create_mesh_curve_num_nodes
INTEGER fem_create_mesh_curve_num_elems
STRING fem_create_mesh_c_nodes_created[VIRTUAL]
STRING fem_create_mesh_c_elems_created[VIRTUAL]
fem_create_mesh_curv_1( "Curve 153:155", 16384, 0.53370899, "Bar2", "#", "#",  @
"Coord 0", "Coord 0", fem_create_mesh_curve_num_nodes,  @
fem_create_mesh_curve_num_elems, fem_create_mesh_c_nodes_created,  @
fem_create_mesh_c_elems_created )
fem_associate_elems_to_ep( "BUStrip", "559:651", 93 )

INTEGER fem_create_mesh_curve_num_nodes
INTEGER fem_create_mesh_curve_num_elems
STRING fem_create_mesh_c_nodes_created[VIRTUAL]
STRING fem_create_mesh_c_elems_created[VIRTUAL]
fem_create_mesh_curv_1( "Curve 168:170", 16384, 0.53517598, "Bar2", "#", "#",  @
"Coord 0", "Coord 0", fem_create_mesh_curve_num_nodes,  @
fem_create_mesh_curve_num_elems, fem_create_mesh_c_nodes_created,  @
fem_create_mesh_c_elems_created )
fem_associate_elems_to_ep( "BLStrip", "652:744", 93 )

$# ----------------------------------------------Mesh Ribs----------------------------------------------

sys_poll_option( 0 )
uil_viewport_post_groups.posted_groups( "default_viewport", 1, ["MRibs"] )
repaint_graphics(  )

INTEGER fem_create_mesh_surfa_num_nodes
INTEGER fem_create_mesh_surfa_num_elems
STRING fem_create_mesh_s_nodes_created[VIRTUAL]
STRING fem_create_mesh_s_elems_created[VIRTUAL]
fem_create_mesh_surf_4( "Paver", 49680, "Surface 1:4", 4, ["0.210624", "0.1",  @
"0.2", "1.0"], "Tria3", "#", "#", "Coord 0", "Coord 0",  @
fem_create_mesh_surfa_num_nodes, fem_create_mesh_surfa_num_elems,  @
fem_create_mesh_s_nodes_created, fem_create_mesh_s_elems_created )
fem_associate_elems_to_ep( "ShellRibs", "745:1090", 346 )

$# ----------------------------------------------Mesh Spars----------------------------------------------

sys_poll_option( 0 )
uil_viewport_post_groups.posted_groups( "default_viewport", 1, ["MSpars"] )
repaint_graphics(  )
```

```
INTEGER fem_create_mesh_surfa_num_nodes
INTEGER fem_create_mesh_surfa_num_elems
STRING fem_create_mesh_s_nodes_created[VIRTUAL]
STRING fem_create_mesh_s_elems_created[VIRTUAL]
fem_create_mesh_surf_4( "Paver", 49680, "Surface 5 6", 4, ["0.197665", "0.1",  @
"0.2", "1.0"], "Tria3", "#", "#", "Coord 0", "Coord 0",  @
fem_create_mesh_surfa_num_nodes, fem_create_mesh_surfa_num_elems,  @
fem_create_mesh_s_nodes_created, fem_create_mesh_s_elems_created )
fem_associate_elems_to_ep( "ShellSpars", "1091:2009", 919 )

$# ----------------------------------------Mesh WB Skin----------------------------------------

sys_poll_option( 0 )
uil_viewport_post_groups.posted_groups( "default_viewport", 1, ["MWB"] )
repaint_graphics(  )

INTEGER fem_create_mesh_surfa_num_nodes
INTEGER fem_create_mesh_surfa_num_elems
STRING fem_create_mesh_s_nodes_created[VIRTUAL]
STRING fem_create_mesh_s_elems_created[VIRTUAL]
fem_create_mesh_surf_4( "IsoMesh", 49152, "Surface 7:24", 1, ["0.215789"],  @
"Tria3", "#", "#", "Coord 0", "Coord 0", fem_create_mesh_surfa_num_nodes,  @
fem_create_mesh_surfa_num_elems, fem_create_mesh_s_nodes_created,  @
fem_create_mesh_s_elems_created )
fem_associate_elems_to_ep( "ShellSkinWB", "2010:5357", 3348 )

$# ----------------------------------------Mesh Skin----------------------------------------

sys_poll_option( 0 )
uil_viewport_post_groups.posted_groups( "default_viewport", 1, ["MSkin"] )
repaint_graphics(  )

INTEGER fem_create_mesh_surfa_num_nodes
INTEGER fem_create_mesh_surfa_num_elems
STRING fem_create_mesh_s_nodes_created[VIRTUAL]
STRING fem_create_mesh_s_elems_created[VIRTUAL]
fem_create_mesh_surf_4( "IsoMesh", 49152, "Surface 25:42", 1, ["0.545808"],  @
"Tria3", "#", "#", "Coord 0", "Coord 0", fem_create_mesh_surfa_num_nodes,  @
fem_create_mesh_surfa_num_elems, fem_create_mesh_s_nodes_created,  @
fem_create_mesh_s_elems_created )
fem_associate_elems_to_ep( "ShellSkin", "5358:13441", 8084 )

$# ----------------------------------------Verify Equivalence----------------------------------------

mesh_seed_display_mgr.erase(  )
REAL fem_equiv_all_x_equivtol_ab
INTEGER fem_equiv_all_x_segment
fem_equiv_all_group4( [" "], 0, "", 1, 1, 0.0049999999, FALSE,  @
fem_equiv_all_x_equivtol_ab, fem_equiv_all_x_segment )
repaint_graphics(  )


$# ----------------------------------------------------Fix----------------------------------------------------

loadsbcs_create2( "Fix", "Displacement", "Nodal", "", "Static", ["Surface 1"], @
 "Geometry", "Coord 0", "1.", ["<0 0 0>", "<0 0 0>", "<    >", "<    >"], [ @
```

```
"", "", "", ""] )


$# ------------------------------------------------Force-----------------------------------------------

loadsbcs_create2( "Force", "Force", "Nodal", "", "Static", ["Point 8"],  @
"Geometry", "Coord 0", "1.", ["< 50 1000 0  >", "< -50 0 -50  >", "<     >", @
 "<    >"], ["", "", "", ""] )

$# ------------------------------------------------Visualization----------------------------------------

uil_viewport_post_groups.posted_groups( "default_viewport", 7, ["Frame",  @
"MRibs", "MSeed", "MSkin", "MSpars", "MStrips", "MWB"] )
ga_group_current_set("Frame")

sys_poll_option( 0 )
uil_toolbar.shaded_smooth(  )
ga_view_aa_set( -168, -10, 165 )
ga_view_zoom_set( 3.5)
repaint_graphics(  )
```