FUN3D v13.4 Training
Session 10: Parameterization Tools

Bill Jones
Setting

• FUN3D shape design relies on a pre-defined relationship between a set of parameters, or design variables, and the discrete surface mesh coordinates

• Given $DV$, surface parameterization determines $X_{surf}$
  • For example, given the current value of wing thickness at a location, what are the corresponding xyz-coordinates of the mesh?

• This narrows down the number of design variables from hundreds of thousands (raw mesh points) to dozens or hundreds
  • Optimizers will perform more efficiently
  • Smoother design space

• An additional requirement of the parameterization package is that it provides the Jacobian of the relationship between the design variables and the surface mesh, $\frac{\partial X_{surf}}{\partial DV}$

• While users may provide their own parameterization scheme, FUN3D is set up to handle three common packages:
  • MASSOUD: Aircraft-centric design variables (thickness, camber, planform, twist, etc)
  • BandAids: General FFD based tool
  • Sculptor®: Commercial package from Optimal Solutions
Learning Goals

- Parameterize geometry with respect to DVs to control shape
  - MASSOUD
  - BandAids
- Generate perturbed surface mesh and SDs for FUN3D design
  - Visual validation
- What we will not cover
  - Body transformations
  - How to use the data in FUN3D
    - That will be covered in the next session
MASSOUD

- **Multidisciplinary Aerodynamic-Structural Shape Optimization Using Deformation**
  - AIAA-2000-4911 (Jamshid Samareh)
- Used to generate consistent models for MDAO
  - Same shape changes communicated across all disciplines
- Highly tailored for aerodynamic shapes
  - Parameters familiar to engineer
- Mesh based parameterization

---

**Multidisciplinary Aero/Structural Shape Optimization Using Deformation (MASSOUD)**
MASSOUD Key Ideas

- Uses soft object animation algorithms for deforming meshes
  - Nonlinear global deformation (twist and dihedral)
  - NURBS surface (camber and thickness)
  - Free-form deformation (planform)
- Parameterizes the discipline meshes
  - Avoids mesh regeneration
- Parameterizes the changes in shape, not the shape itself
  - No need to reproduce shape
    - Reduces the number of design variables
MASSOUD Twist and Shear

- Nonlinear Global Deformation
  - Wrapped in twist cylinder
    - Twisted and sheared in planes along span normal to twist vector

Twist parameterization of a generic wing

Twist parameterization of a generic transport

Extreme deformation of a generic transport
MASSOUD Camber and Thickness

- Non-Uniform Rational B-Spline (NURBS)
  - Represents the shape changes not the shape

NURBS Control Points for Camber and Thickness

Camber

Thickness

Extreme Camber and Thickness deformation
MASSOUD Planform

- Free-form Deformation (FFD)
  - Surround shapes with quadrilaterals
MASSOUD Installation

- Distributed as source code
  - Single `Makefile` uses GNU C compiler (`gcc`)
    - Any localization must be done manually
  - Creates two executables
    - `massoudDesignDriver` creates parameterization
    - `massoud` surface mesh perturbation with sensitivity data
MASSOUD Process

Preprocessing Phase

Step 1: Determine # and locations of design variables

Baseline Model

Step 2: Create baseline analysis mesh for discipline $N$

Mesh $N$

Step 3: Parameterize each discipline

massoudDesignDriver

massoudBaselineModel

gpNFile

Step 4: Perturb and compute sensitivity

massoud

SD Input

Design Group

Design Locations

Design Variable Templates

Processing Phase

Design Variable

Plot file (Tecplot™)

New Mesh $N$

New SDs Mesh $N$

Plot file (Tecplot™)
MASSOUD Step 1

- Parameterization requires input to define DV locations
  - Small ASCII file
  - Contains 7 groups of oriented curves
    - X axis is positive downstream
    - Y is positive out the wing span
      - Y should be positive with curves monotonically increasing
  - GridTool can be used to create the file
### Design location file Case Name Title (SECTION 1)

<table>
<thead>
<tr>
<th>np</th>
<th>ne</th>
<th>ntwist</th>
<th>ncmax</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>1</td>
<td>2</td>
<td>100</td>
<td>0 1 2</td>
</tr>
</tbody>
</table>

### Planform

### Twist vector and Twist

### Leading and Trailing Edges

### Thickness

### Camber

---

**Planform**

**Twist vector and Twist**

**Leading and Trailing Edges**

**Thickness**

**Camber**
1. Planform
   • Cover planform with 5 point quadrilaterals
     • Closed but orientation does not matter
   • 1 Curve per planform section
   • GridTool Family name “planform”
MASSOUD Design Locations

2. Leading Edge
   • Create an $n$ point PWL curve defining the leading edge
     • Must bound all mesh nodes
     • May extend beyond actual geometry
   • GridTool Family name “le”

3. Trailing Edge
   • Create an $n$ point PWL curve defining the trailing edge
     • Must bound all mesh nodes
     • GridTool Family name “te”
MASSOUD Design Locations

4. Twist Vector
   • Create a 2 point curve to represent the twist vector
     • Twist sections defined normal to this vector
     • GridTool Family name “twistv”

5. Twist Location
   • Create an $n$ point PWL curve to represent the $n$ twist locations
   • Airfoil sections defined at these points normal to “twistv”
     • First and last section must bound the Y coordinates of the target mesh
     • GridTool family name “twist”
MASSOUD Design Locations

6. Thickness
   • Chordwise
     • Create an n point PWL curve to represent the n chordwise thickness locations
     • Start, length, and %
     • GridTool family name “tx”
   • Spanwise
     • Create an m point PWL curve to represent the m spanwise thickness locations
     • Should bound Y values of all target mesh nodes
     • Beginning and ending Y coordinates must be bounded by the Y coordinates of both the “le” and “te” curves
     • May be a duplicate of the “twist” curve
     • GridTool family name “ty”
   • n x m set of DVs
MASSOUD Design Locations

7. Camber
   • Same as for Thickness but with GridTool family names “cx” and “cy” respectively
   • May be duplicates of “tx” and “ty”
   • Two curves define $n \times m$ set of DVs
MASSOUD Step 2

- Dump out surface meshes of interest in a Tecplot™ format
  - Includes the surface node coordinates
  - Global ID of the surface nodes wrt the volume mesh
  - FUN3D flow solver CLO ‘--write_massoud_file’
    - Produces “[project]_massoud_bodyN.dat” file for body N
  - Default extracts all viscous boundary surfaces as separate bodies

- FUN3D Namelist controls

```python
&massoud_output
  n_bodies = 2    ! Parameterize 2 bodies
  nbndry(1)      = 6    ! 1st body has 6 boundaries
  boundary_list(1) = ‘3-8’  ! Boundaries in 1st body
  nbndry(2)      = 3    ! 2nd body has 3 boundaries
  boundary_list(2) = ‘9,10,12’  ! Boundaries in 2nd body
/
```

- `boundary_list()` indices should reflect boundary lumping
MASSOUD Step 3

• Generate geometry parameterization

```bash
% massoudDesignDriver -t input_massoud_bndry1.dat \
  designLocations \ 
  design.gp.1
```

• Geometry parameterization is output in “design.gp.1”
  • Used as input to `massoud`

• Additional output
  • “designVariableTemplate”
    • Reference for “design.1” file with zero perturbations
  • “designTemplate.usd”
    • Reference for “design.usd.1” user defined variable links
  • “designVariableTemplateNumber”
    • Lists the DV indices by DV type (planform, twist, etc.)
  • “baselineShape.plt”
    • Tecplot™ readable zero perturbation reference
  • Errors in “GP.log”
MASSOUD Step 4

• Mesh deformation % massoud massoud.N
  • Where MASSOUD input is in “massoud.N”
  • FUN3D design will utilize “customDV.N” for perturbations

```
#MASSOUD INPUT FILE
# Option (0 analysis), (> 0 sd using user dvs ) (-1, sd using massoud dvs)
-1
# core (0 incore solution) (1 out of core solution)
0
# input parameterized file
design.gp.1
# design variable input file
design.1
# input sensitivity file (used for Option > 0)
design.usd.1
# output file mesh file
new1.plt
# output tecplot file for viewing
model.tec.1
# file containing the design variables group
designVariableGroups.1
# user design variable file
[customDV.1]
```
MASSOUD Results

• Visual inspection
  • Tecplot™
    • “model.tec.1.sd1” contains mesh and SDs
      • (e.g. XD1, YD1, ZD1… XDndv, YDndv, ZDndv)
  • GridTool
    • Sliders to interactively perturb DVs
    • Twist is non-linear and is only indication of change
What Could Go Wrong (1 of 2)

- Failure … check “GP.log”
- Design locations must be defined to bound all target mesh nodes

Twist Vector

Twist Vector
What Could Go Wrong (2 of 2)

- Design locations must be defined to bound all target mesh nodes

"cy" does not bound tip nodes with full precision
MASSOUD User Defined Variables

- New variables as linear combination of MASSOUD variables

\[
\frac{\partial R}{\partial V_j} = \frac{\partial R}{\partial P_i} \frac{\partial P_i}{\partial V_j}
\]

\[\begin{bmatrix}
\frac{\partial P_1}{\partial V_1} & \frac{\partial P_1}{\partial V_2} & \cdots & \frac{\partial P_{i_{\text{max}}}}{\partial V_1} \\
\frac{\partial P_1}{\partial V_2} & \frac{\partial P_2}{\partial V_2} & \cdots & \frac{\partial P_{i_{\text{max}}}}{\partial V_2} \\
\vdots & \vdots & \ddots & \vdots \\
\frac{\partial P_1}{\partial V_{j_{\text{max}}}} & \frac{\partial P_2}{\partial V_{j_{\text{max}}}} & \cdots & \frac{\partial P_{i_{\text{max}}}}{\partial V_{j_{\text{max}}}}
\end{bmatrix}
\]

\[V_j\] MASSOUD Design Variables

\[P_i\] User - Defined Design Variables

\[P_1 = V_{10} - V_1\text{ (Chord)}\]
\[P_2 = (V_{10} + V_1)/2\text{ (Mid - Chord Location)}\]
\[P_3 = V_2 = V_{11}\]

\[
\begin{array}{ccc}
P_1 & P_2 & P_3 \\
V_1 & -1 & 0.5 & 0 \\
V_2 & 0 & 0 & 1 \\
V_{10} & 1 & 0.5 & 0 \\
V_{11} & 0 & 0 & 1 \\
\end{array}
\]

M6.usd
# MASSOUD Pros and Cons

**Pros**
- Consistent Meshes
- No need for mesh generation
- Easy to setup (hours)
- Parameterization is fast
- Analytic sensitivity
- Compact set of DVs
- Suitable for high- and low-fidelity application

**Cons**
- Limited to small shape changes
- Fixed topology
- No built-in geometry constraints
- No direct CAD connection
BandAids

- Aerodynamic Shape Parameterization based on Free-Form Deformation
- General application based on free-form deformation
  - Handles complex shapes
  - DVs are not classic aerodynamic parameters
BandAids Key Ideas

1. Parameterize surface mesh
   • Avoids mesh regeneration

2. Use FFD to represent shape perturbations
   • Automates surface parameterization

3. Parameterize changes in shape perturbation, not the shape itself
   • Reduces the number of design variables
BandAids FFD (1 of 3)

- Based on algorithm used in computer animation
  - Control points are DVs
  - Immersed in Jell-O
- Design variables have no aerodynamic significance
  - Only those near surface have significant impact
BandAids FFD (2 of 3)

- Many more control points in 3D
  - Only those near surface have impact on surface
BandAids FFD (3 of 3)

- Equivalent 3D bi-variant form of tri-variant FFD
  - Collapse CPs onto surface
    - Move CP moves surface underneath
  - Number of DVs reduced from $N^3$ to $N^2$
  - 4 sided Bandaid marking surface over geometry
    - Moves only surface to which it is collapsed
    - No MDO
BandAids Parameterizes Changes

- Shape changes are small
  - Can be represented with fewer CPs than surface
- Maintains surface mesh character/quality

\[ r_n(v) = r^n_b + \Delta r_n(v) \]

- Surface mesh point
- Design variable vector
- Baseline surface mesh
- Shape changes

NURBS control points for camber & thickness
BandAids Installation

• Distributed as source code
  • Single **Makefile** uses GNU C compiler (**gcc**)
    • Any localization must be done manually
  • Creates a single executable
    • `bandAids` parameterization and deformation
BandAids Marking Surfaces (1 of 2)

- Create structured marking surface
  - Marks portion of geometry to parameterize
  - Can span multiple geometry surfaces

![Diagram of marking surfaces and mesh](image)
BandAids Marking Surfaces (2 of 2)

- Marking surface interpolated by reference with $n \times m$ CPs
  - $n \times m$ DVs
BandAids Execution

% bandAids inMesh.plt \
inDesignSurf.p3d \output \numDesignInU \numDesignInV \[tol]

• “inMesh.plt” target mesh in Tecplot™ format
• “inDesignSurf.p3d” marking surface in PLOT3D format
• “outfile” output file name prefix
• “numDesignInU” number of design variables in U-direction
• “numDesignInV” number of design variables in V-direction
• “tol” optional, max gap between mesh and marking surface
• User defined variables are created if a “bandaids.usd” file exists at execution
BandAids Output

- Execution produces seven files:
  - "output.bandaid"
    - All non-zero shape information
    - Read directly by FUN3D
  - "output.distance.plt"
    - Tecplot™ file with the surface mesh including the distance between the surface mesh and marking surface
  - "output.distanceSD.plt"
    - Tecplot™ file containing surface mesh and sensitivity data
  - "bandAidsSample.dvs"
    - Template for input design variable file
  - "bandAidsAll.usd", "bandAidsCol.usd", and "bandAidsRow.usd"
    - Templates to base "bandaids.usd" used for DV linking
    - Requires a subsequent `bandaids` run for linked variables
BandAids Deformation

• Not necessary with FUN3D as all deformation is linear
  • Useful for validation
• Execute `bandAids` with `-deformMesh`
  
  ```
  % bandAids -deformMesh \
  output.distanceSD.plt \ 
  my.dvs \ 
  new.plt
  
  ``
• “`output.distanceSD.plt`”
  • Tecplot™ file containing surface mesh and sensitivity data
• “`my.dvs`”
  • Input DV perturbations
• “`new.plt`”
  • Deformed surface mesh
BandAids Results

• Visual inspection
  • Tecplot™
    • “output.distanceSD.plt” contains mesh and SDs
      • (e.g. XD1, YD1, ZD1… XDndv, YDndv, ZDndv)
  • GridTool
    % GridTool -d output.distanceSD.plt
    • Sliders to interactively perturb DVs
# BandAids Pros and Cons

<table>
<thead>
<tr>
<th>Pros</th>
<th>Cons</th>
</tr>
</thead>
<tbody>
<tr>
<td>• General Application</td>
<td>• Non-intuitive DVs</td>
</tr>
<tr>
<td>• Consistent Meshes</td>
<td>• Limited to small shape changes</td>
</tr>
<tr>
<td>• No need for mesh generation</td>
<td>• No built-in geometry constraints</td>
</tr>
<tr>
<td>• Easy to setup (hours)</td>
<td>• No direct CAD connection</td>
</tr>
<tr>
<td>• Parameterization is fast</td>
<td></td>
</tr>
<tr>
<td>• Analytic sensitivity</td>
<td></td>
</tr>
<tr>
<td>• Compact set of DVs</td>
<td></td>
</tr>
<tr>
<td>• Suitable for high- and low- fidelity application</td>
<td></td>
</tr>
</tbody>
</table>
What We Learned

- MASSOUD parameterizes with aerodynamic parameters
  - Best applied to aerodynamic shapes
- BandAids provides general application
  - Albeit w/o intuitive parameters
- Both mesh based parameterization
- Both tools parameterize shape changes not shape
  - Reduces number of DVs
- Both provide mesh perturbation with SDs suitable for FUN3D