

Computational Aircraft Prototype Syntheses



Training Session 7

Meshing for CFD: AFLR

ESP v1.22

Marshall Galbraith

galbramc@mit.edu

Massachusetts Institute of Technology

Bob Haines

haines@mit.edu

John F. Dannenhoffer, III

jfdannen@syr.edu

Syracuse University

- AFLR4 surface mesh generation
 - Mesh length scaling (`capsMeshLength`)
 - Edge spacing controls
 - Proximity Detection
- AFLR3 volume mesh generation
 - Inviscid mesh generation (linking `Surface_Mesh`)
 - Viscous boundary layer mesh generation
- Suggested Exercises

Advancing-Front/Local-Reconnection Grid Generator

- AFLR mesh generator suite by Prof. David Marcum at Mississippi State
 - AFLR2 – 2D unstructured meshes
 - AFLR3 – 3D unstructured meshes with boundary layers
 - AFLR4 – CAD surface meshes
- AFLR2/AFLR3 developed over past decades
- AFLR4 extensive development as part of CAPS project
 - Actively under development

AFLR4/AFLR3 AIM Documentation

- AFLR generates meshes in memory
- geometry.view shows surface tessellation after runAnalysis

session07/01_aflr4_InviscidWing.py

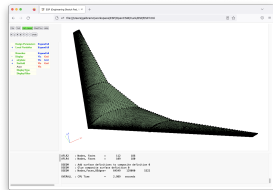
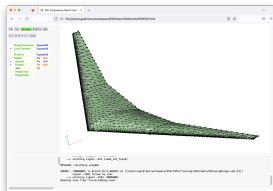
```
# Create aflr4 aim
aflr4 = capsProblem.analysis.create(aim = "aflr4AIM",
                                    name = "aflr4")

# View the bodies
aflr4.geometry.view()

# Mark capsMesh == Farfield with a Farfield bcType
aflr4.input.Mesh_Sizing = {"Farfield": {"bcType": "Farfield"}}

# Run AIM analysis
aflr4.runAnalysis()

# View the surface tessellation
aflr4.geometry.view()
```



- Farfield boundary set via “Mesh_Sizing”, or tagged with
`ATTRIBUTE AFLR_GBC $FARFIELD_UG3_GBC`
- “Mesh_Sizing” via capsMesh attributes on FACE/EDGE/NODE

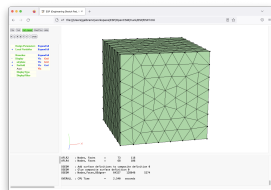
session07/02_aflr4_Farfield.py

```
# Mark capsMesh == Farfield with a Farfield bcType
aflr4.input.Mesh_Sizing = {"Farfield": {"bcType": "Farfield"}}

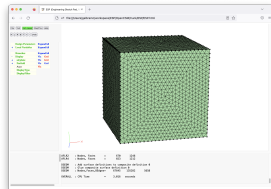
# Farfield growth factor
aflr4.input.ff_cdfc = 1.4

# Run AIM
aflr4.runAnalysis()

# View the surface tessellation
aflr4.geometry.view()
```



aflr4.input.ff_cdfc = 1.4



aflr4.input.ff_cdfc = 1.1

- AFLR4 reference length (ref_len) bounds element sizes
- Rule of thumb: characteristic length of geometry
 - Mean Aerodynamic Chord, diameter of fuselage, etc.
- CAPS: reference length computed from capsMeshLength

ATTRIBUTE capsMeshLength wing:mac

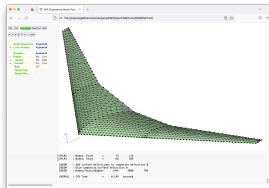
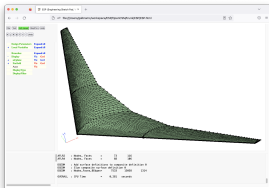
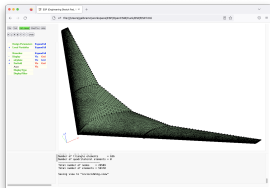
session07/03_aflr4_MeshLength.py

```
# Scaling factor to compute AFLR4 'ref_len' parameter via
# ref_len = capsMeshLength * Mesh_Length_Factor
aflr4.input.Mesh_Length_Factor = 5

# Relative scale of maximum spacing bound relative to ref_len
# max_spacing = max_scale * ref_len
aflr4.input.max_scale = 0.1

# Relative scale of minimum spacing bound relative to ref_len
# min_spacing = min_scale * ref_len
aflr4.input.min_scale = 0.01

# Absolute scale of minimum spacing bound for proximity
# abs_min_spacing = abs_min_scale * ref_len
aflr4.input.abs_min_scale = 0.001
```



session07/03_aflr4_MeshLength.py

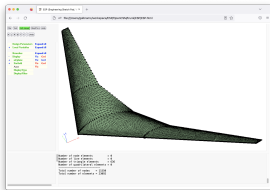
```
# Use mesh length factor to make a series of meshes (not a family)
for Length_Factor in [1, 3, 9]:
    aflr4.input.Mesh_Length_Factor = Length_Factor

    # Run AIM
    aflr4.runAnalysis()

    # View the surface tessellation
    aflr4.geometry.view()
```

- Increase resolution of “sharp” edges (e.g. trailing edges)
- Set via “Mesh_Sizing”, or attribute applied to FACES

ATTRIBUTE AFLR4_Edge_Scale_Factor_Weight 1



session07/04_aflr4_EdgeWeight.py

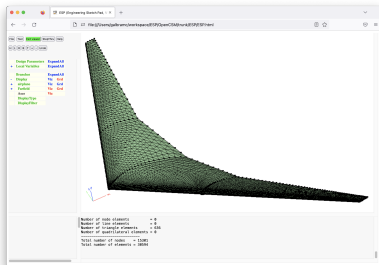
```
# Edge mesh spacing can be scaled on surfaces based on
# discontinuity level between adjacent surfaces on both sides of the edge.
# The level of discontinuity potentially reducing the edge spacing.
# The edgeWeight scale factor weight is used as an interpolation weight
# between the unmodified spacing and the modified spacing.
aflr4.input.Mesh_Sizing = {"leftWing": {"edgeWeight":1.0},
                           "Farfield": {"bcType": "Farfield"}}
```

- Increase/Decrease resolution on FACE/EDGE
- Set via “Mesh_Sizing”, or attribute applied FACE/EDGE

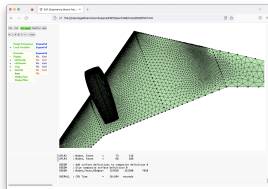
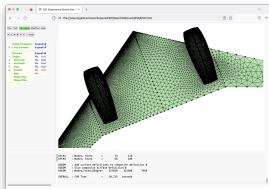
ATTRIBUTE AFLR4_Scale_Factor 0.25

session07/05_aflr4_ScaleFactor.py

```
# Apply a local scaling factor to capsMesh="leftWing"  
aflr4.input.Mesh_Sizing = {"riteWing": {"scaleFactor":2.0},  
                           "leftWing": {"scaleFactor":0.5},  
                           "Farfield": {"bcType":"Farfield"}}
```

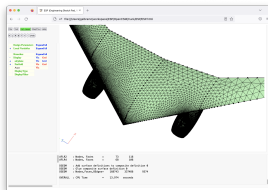
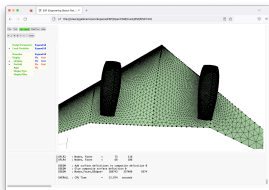


- Proximity detection imprints meshes from close bodies on each other
- Wing coarsened to illustrate imprinting
- Individual bodies treated as separate components



session07/06_aftr4_Proximity.py

```
# Set mesh sizing parameters
aftr4.input.Mesh_Sizing = {"Farfield": {"bcType": "Farfield"},
                           "leftWing": {"scaleFactor": 50},
                           "Nacelle" : {"scaleFactor": 0.5}}
```



- AFLR4_Cmp_ID FACE attribute distinguishes components

ESP/viewCFDViscous.udc

```
SET AFLR4_Cmp_Fuse 1
SET AFLR4_Cmp_Wing 2
SET AFLR4_Cmp_Htail 3
SET AFLR4_Cmp_Vtail 4
SET AFLR4_Cmp_Nacelle 5
SET AFLR4_Cmp_Pylon 6
SET AFLR4_Cmp_Farfield 7
```

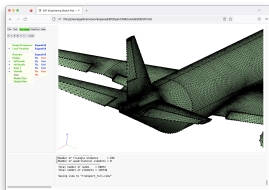
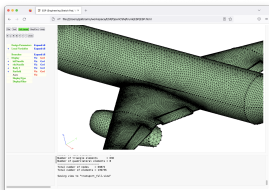
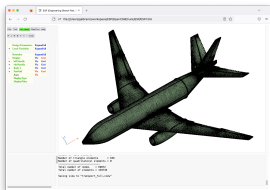
```
FACE HAS tagComp=leftWing tagType=upper
SET capsGroup=Wing
SET capsBound=upperWing
SET capsMesh=leftWing
SET AFLR4_Cmp_ID=!AFLR4_Cmp_Wing
```

```
FACE HAS tagComp=leftNacelle
SET capsGroup=Nacelle
SET capsMesh=Nacelle
SET AFLR4_Cmp_ID=!AFLR4_Cmp_Nacelle
```

Execute: session07/07_aflr4_ProximityComponents.py

- Inviscid surface mesh for the full transport configuration
 - ~ 10 s
 - 70k Nodes
 - 140k Triangles

Execute: session07/08_aftr4_InviscidTransport.py



- AFLR4 surface mesh generation
 - Mesh length scaling (`capsMeshLength`)
 - Edge spacing controls
 - Proximity Detection
- AFLR3 volume mesh generation
 - Inviscid mesh generation (linking `Surface_Mesh`)
 - Viscous boundary layer mesh generation
- Suggested Exercises

Surface Mesh Transfer

- AFLR4 AIM output Surface_Mesh is linked to AFLR3 AIM input Surface_Mesh
- Transfers surface mesh from AFLR4 to AFLR3

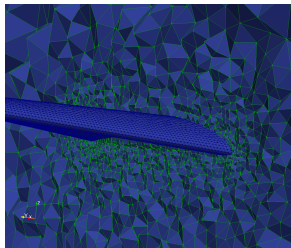
session07/09_aflr4_aflr3_InviscidWing.py

```
# Create AFLR3 AIM to generate the volume mesh
aflr3 = capsProblem.analysis.create(aim = "aflr3AIM",
                                   name = "aflr3")

# Link the aflr4 Surface_Mesh as input to aflr3
aflr3.input["Surface_Mesh"].link(aflr4.output["Surface_Mesh"])

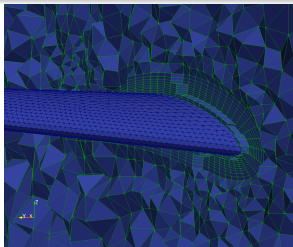
# Dump VTK files for visualization
aflr3.input.Proj_Name = "TransportWing"
aflr3.input.Mesh_Format = "VTK"

# Run AIM
aflr3.runAnalysis()
```



- Boundary layer meshes are required for viscous CFD
- AFLR3 "grows" boundary layer meshes from viscous surfaces

- Global boundary layer parameter¹
(automatically ignores Farfield)
- Mesh_Gen_Input_String to add any
AFLR3 command line flags



session07/10_aflr4_aflr3_ViscousWing.py

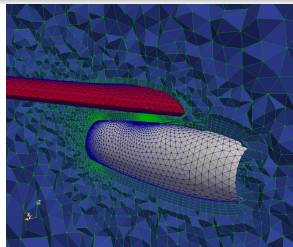
```
# Specify boundary layer maximum layers.
# Initial spacing and minimum thickness are scaled by capsMeshLength
aflr3.input.BL_Max_Layers      = 10
aflr3.input.BL_Initial_Spacing = 0.01
aflr3.input.BL_Thickness      = 0.1

# Specify prism boundary layer elements
aflr3.input.Mesh_Gen_Input_String = "-blc"
```

¹NOTE: Unreasonably coarse boundary layers in examples

- Boundary layer meshes are required for viscous CFD
- AFLR3 "grows" boundary layer meshes from viscous surfaces

- capsMesh wise boundary layers parameters¹
- Properly treats colliding layers



session07/11_aflr4_aflr3_ViscousWingPod.py

```
# Specify boundary layer maximum layers.
```

```
aflr3.input.BL_Max_Layers = 10
```

```
# Specify prism boundary layer elements
```

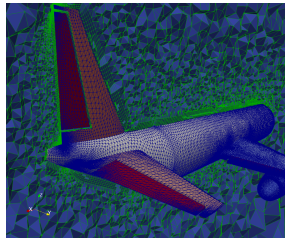
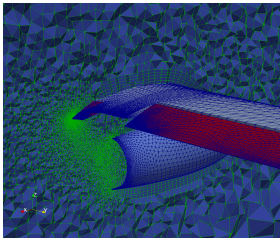
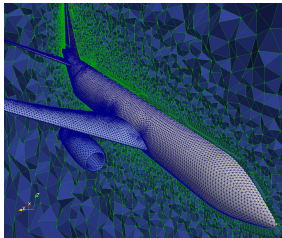
```
aflr3.input.Mesh_Gen_Input_String = "-blc"
```

```
# Set mesh sizing parameters
```

```
aflr3.input.Mesh_Sizing = {"leftWing" : {"boundaryLayerSpacing":0.01, "boundaryLayerThickness":0.1},  
                           "riteWing" : {"boundaryLayerSpacing":0.01, "boundaryLayerThickness":0.1},  
                           "Nacelle"  : {"boundaryLayerSpacing":0.02, "boundaryLayerThickness":0.1}}
```

- Viscous mesh for full transport configuration¹
 - ~1.0 min
 - 380k Nodes
 - 1.8M Elements

Execute: `session07/12_afmr4_afmr3_ViscousTransport.py`



¹NOTE: Unreasonably coarse boundary layers in examples

curv_factor

- Explore the impact of AFLR4 “curv_factor”

Inviscid Transport

- Make the surface mesh finer for the engine pods of the InviscidTransport
- Coarsen the fuselage surface mesh for the InviscidTransport without coarsening the wing/tail surfaces

Inviscid Mesh Sequence

- For the InviscidTransport, generate surface meshes with approximate element counts of:
 - 150,000
 - 250,000
 - 300,000
- Create your own (optionally share it galbramc@mit.edu)