AFLR4 Analysis Interface Module (AIM)

Ryan Durscher
AFRL/RQVC

May 14, 2018

Contents

1 Introduction 1

1.1 AFLR4 AIM Overview 1

2 Geometry Representation and Analysis Intent 2

3 AIM Inputs 2

4 AIM Shareable Data 3

5 AIM Outputs 3

6 Mesh Sizing 3

6.1 JSON String Dictionary 3

6.2 Single Value String 4

Bibliography 5

1 Introduction

1.1 AFLR4 AIM Overview

A module in the Computational Aircraft Prototype Syntheses (CAPS) has been developed to interact with the unstructured, surface grid generator AFLR4 [2] [1].

The AFLR4 AIM provides the CAPS users with the ability to generate "unstructured, 3D surface grids" using an "Advancing-Front/Local-Reconnection (AFLR) procedure." Both triangular and quadrilateral elements may be generated.

An outline of the AIM's inputs and outputs are provided in AIM Inputs and AIM Outputs, respectively.

The accepted and expected geometric representation and analysis intentions are detailed in Geometry Representation and Analysis Intent.

Details of the AIM's shareable data structures are outlined in AIM Shareable Data if connecting this AIM to other AIMs in a parent-child like manner.

Example surface meshes:

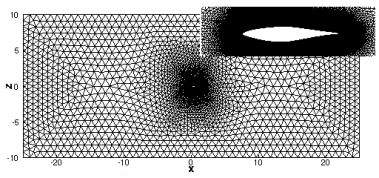

Figure 1: AFLR4 meshing example - 2D Airfoil

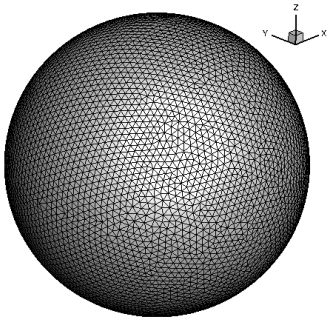

Figure 2: AFLR4 meshing example - Sphere

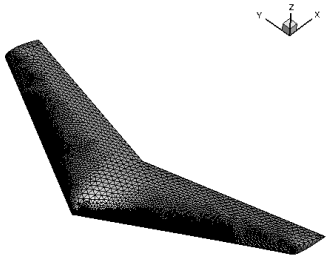

Figure 3: AFLR4 meshing example - Airfoil

2 Geometry Representation and Analysis Intent

The geometric representation for the AFLR4 AIM requires the body(ies) be either face body(ies) (FACEBODY), solid body(ies) (SOLIDBODY) or non- and manifold sheet body(ies) (SHEETBODY). Furthermore, the attribute capsIntent should be set to CFD and STRUCTURE analyses, or ALL.

3 AIM Inputs

The following list outlines the AFLR4 meshing options along with their default value available through the AIM interface.

- **Proj_Name = NULL**
  This corresponds to the output name of the mesh. If left NULL, the mesh is not written to a file.
- **Tess_Params = [0.025, 0.001, 15.0]**
  Body tessellation parameters. Tess_Params[0] and Tess_Params[1] get scaled by the bounding box of the body. (From the EGADS manual) A set of 3 parameters that drive the EDGE discretization and the FACE triangulation. The first is the maximum length of an EDGE segment or triangle side (in physical space). A zero is flag that allows for any length. The second is a curvature-based value that looks locally at the deviation between the centroid of the discrete object and the underlying geometry. Any deviation larger than the input value will cause the tessellation to be enhanced in those regions. The third is the maximum interior dihedral angle (in degrees) between triangle facets (or Edge segment tangents for a WIREBODY tessellation), note that a zero ignores this phase.
- **Mesh_Quiet_Flag = False**
  Complete suppression of mesh generator (not including errors)

- **Mesh_Format = "AFLR3"**
  Mesh output format. Available format names include: "AFLR3", "VTK", "TECPLOT", "STL" (quadrilaterals will be split into triangles), "FAST".

- **Mesh_ASCII_Flag = True**
  Output mesh in ASCII format, otherwise write a binary file if applicable.

- **Mesh_Gen_Input_String = NULL**
  Meshing program command line string (as if called in bash mode). Use this to specify more complicated options/use features of the mesher not currently exposed through other AIM input variables. Note that this is the exact string that will be provided to the volume mesher; no modifications will be made. If left NULL an input string will be created based on default values of the relevant AIM input variables.

- **Edge_Point_Min = 2**
  Minimum number of points along an edge to use when creating a surface mesh.

- **Edge_Point_Max = NULL**
  Maximum number of points along an edge to use when creating a surface mesh.

- **Mesh_Sizing = NULL**
  See Mesh Sizing for additional details.

4 AIM Shareable Data

The AFLR4 AIM has the following shareable data types/values with its children AIMs if they are so inclined.

- **Surface_Mesh**
  The returned surface mesh after AFLR4 execution is complete in meshStruct (see meshTypes.h) format.

- **Attribute_Map**
  An index mapping between capsGroups found on the geometry in mapAttrToIndexStruct (see miscTypes.h) format.

5 AIM Outputs

The following list outlines the AFLR4 AIM outputs available through the AIM interface.

- **Done** = True if a surface mesh was created on all surfaces, False if not.

6 Mesh Sizing

Structure for the mesh sizing tuple = ("CAPS Group Name", "Value"). "CAPS Group Name" defines the capsGroup on which the sizing information should be applied. The "Value" can either be a JSON String dictionary (see Section JSON String Dictionary) or a single string keyword string (see Section Single Value String)

6.1 JSON String Dictionary

If "Value" is a JSON string dictionary (e.g. "Value" = {"edgeDistribution": "Even", "numEdgePoints": 100}) the following keywords (= default values) may be used:

- **edgeDistribution = "Even"**
  Edge Distribution types. Options: Even (even distribution), Tanh (hyperbolic tangent distribution).

- **numEdgePoints = 0**
  Number of points along an edge.

- **initialNodeSpacing = [0.0, 0.0]**
  Initial (absolute) node spacing along an edge.

- **tessParams = (no default)**
  Face tessellation parameters, example [0.1, 0.01, 20.0]. (From the EGADS manual) A set of 3 parameters that drive the EDGE discretization and the FACE triangulation. The first is the maximum length of an EDGE segment or triangle side (in physical space). A zero is flag that allows for any length. The second is a curvature-based value that looks locally at the deviation between the centroid of the discrete object and the underlying geometry. Any deviation larger than the input value will cause the tessellation to be enhanced in those regions. The third is the maximum interior dihedral angle (in degrees) between triangle facets (or Edge segment tangents for a WIREBODY tessellation), note that a zero ignores this phase.

6.2 Single Value String

If "Value" is a single string, the following options maybe used:

- (NONE Currently)

References

[1] David L Marcum. Unstructured grid generation using automatic point insertion and local reconnection. *The Handbook of Grid Generation*, pages 18–1, 1998. 1

[2] David L. Marcum and Nigel P. Weatherill. Unstructured grid generation using iterative point insertion and local reconnection. *AIAA Journal*, 33(9):1619–1625, Sep. 1995. 1