

CAPS Example AIM for AVL

Bob Haimes

March 13, 2015

1 Goals

The use of lower-dimensional design tools is clearly desirable in a multidisciplinary/multi-fidelity aero design optimization setting. This is the crux of the Computational Aircraft Prototype Syntheses (CAPS) program. In many ways describing geometry appropriate for AVL (the Athena Vortex Lattice) code is more cumbersome than higher fidelity codes that require an Outer Mold Line.

The goal is to make a CAPS AIM (Analysis Input Module) that directly feeds input to AVL and extracts the output quantities of interest from AVL's execution. This needs to be consistent with a build description that is hierarchical and multi-fidelity. That is, the build description that generates the geometric data at this level can be further enhanced to produce the complete OML of the aircraft design under consideration.

As for the geometric description, AVL requires airfoil section data specified at the appropriate locations that describe the *skeleton* of the aircraft. These sections when *lofted* as groups and finally *unioned* together builds the OML. Clearly, intercepting the state of the geometry before these higher-level operations are applied provides the data appropriate for AVL. This naturally constructs a hierarchal geometric view where a design can progress into higher fidelities and feedback can be achieved where we can go back to this level of description when need be.

2 Assumptions

The AVL coordinate system assumption (X – downstream, Y – out the right wing, Z – up) needs to be followed.

Within **OpenCSM** there are a number of airfoil generation UDPs (User Defined Primitives). These include NACA 4 series, a more general NACA 4/5/6 series generator, Sobieczky's PARSEC parameterization and Kulfan's CST parameterization. All of these UDPs generate **EGADS** *FaceBodies* where the *Face*'s underlying *Surface* is planar and the bounds of the *Face* is a closed set of *Edges* whose underlying *Curves* contain the airfoil shape. In all cases there is a *Node* that represents the *Leading Edge* point and one or two *Nodes* at the *Trailing Edge* – one if the representation is for a sharp TE and the other if the definition is open or blunt. If there are 2 *Nodes* at the back, then there are 3 *Edges* all together and closed, even though the airfoil definition was left open at the TE. All of this information will be used to automatically fill in the AVL geometry description.

It should be noted that general construction in either **OpenCSM** or even **EGADS** will be supported as long as the topology described above is used. But care should be taken when constructing the airfoil shape so that a discontinuity (i.e., simply C^0) is not generated at the *Node* representing the *Leading Edge*. This can be done by splining the entire shape as one and then intersecting the single *Edge* to place the LE *Node*.

The rest of the information and options required to fill out the AVL geometry input file (**xxx.avl**) will be found in the attributes attached to the *FaceBody* itself. The conventions used will be described in the next section.

3 Attribute Use for the construction of xxx.avl

The following list of attributes drives the AVL geometric definition. Each *FaceBody* which relates to AVL Sections will be marked up in an appropriate manner to drive the input file construction. Many attributes are required and those that are optional are marked so in the description:

- **capsFidelity.** This attribute is a CAPS requirement to indicate the *Body's* fidelity and in this case should refer to AVL-like usage.
- **avlSurface.** This string attribute labels the *FaceBody* as to which AVL Surface the section is assigned. This should be something like: *Main_Wing*, *Horizontal_Tail*, and etc.
This informs the AVL AIM to collect all *FaceBodies* that match this attribute into a single AVL Surface.
- **avlComponent** [optional]. This integer (or real) attribute is examined for the AVL Component index. This may be on any or all *FaceBodies* associated with the AVL Surface.
- **avlKeyword** [optional]. This string attribute may contain the word (or words): NOWAKE, NOALBE and/or NOLOAD. Multiple keywords need to be delimited by a colon (':'). This is associated with the AVL Surface.
- **avlSecValues.** This real valued attribute must be of at least 5 *doubles* in length and contain:
 1. **Nchord** – The number of chordwise horseshoe vortices placed on the surface.
 2. **Cspace** – The chordwise vortex spacing parameter.
 3. **Nspan** – The number of spanwise horseshoe vortices placed on the surface.
 4. **Sspace** – The spanwise vortex spacing parameter.
 5. **iYdup** – The Y symmetry for this Surface (0 – do nothing, 1 – duplicate about the Y=0.0 plane).

It should be noted that the first 2 (and last) values refer to the entire AVL Surface and therefore the first occurrence of the **avlSurface** attribute in the list of *bodies* sets these values in the AVL input file (though **iYdup** can be set in any Section).

- **avlCntrlName** [optional]. This optional numerical attribute indicates that this *FaceBody* Section is a member of the control surface named as part of the attribute name (for example: **avlCntrlAileron** specifies that this Section is part of the *Aileron* control surface). This real valued attribute must be of at least 6 *doubles* in length and contain:
 1. **gain** – The control deflection gain.
 2. **Xhinge** – The x/c location of hinge.
 3. **Xhvec** – The first component of the vector giving hinge axis about which surface rotates.
 4. **Yhvec** – The second component.
 5. **Zhvec** – The third component.
 6. **SgnDup** – The signed magnitude (or simply the scale factor) for the deflection of the duplicated surface.

The AVL Sections are automatically generated, one from each *FaceBody* and the details extracted from the geometry. **Xle**, **Yle**, and **Zle**, are taken from the *Node* Associated with the *Leading Edge*. The **Chord** is computed by getting the distance between the LE and TE (if there are 3 *Edges* in the *FaceBody* the TE point is considered the mid-position on that third *Edge*). **Ainc** is computed by registering the chordal direction of the *FaceBody* against the X-Z plane. The airfoil shapes are generated by sampling the *Curves* and put directly in the input file via the **AIRFOIL** keyword after being normalized.

Also note that this first implementation is not intended to provide complete control over AVL. In particular, there is no mention above of the **BODY**, **DESIGN**, **CLAF**, or **CDCL** AVL keywords.