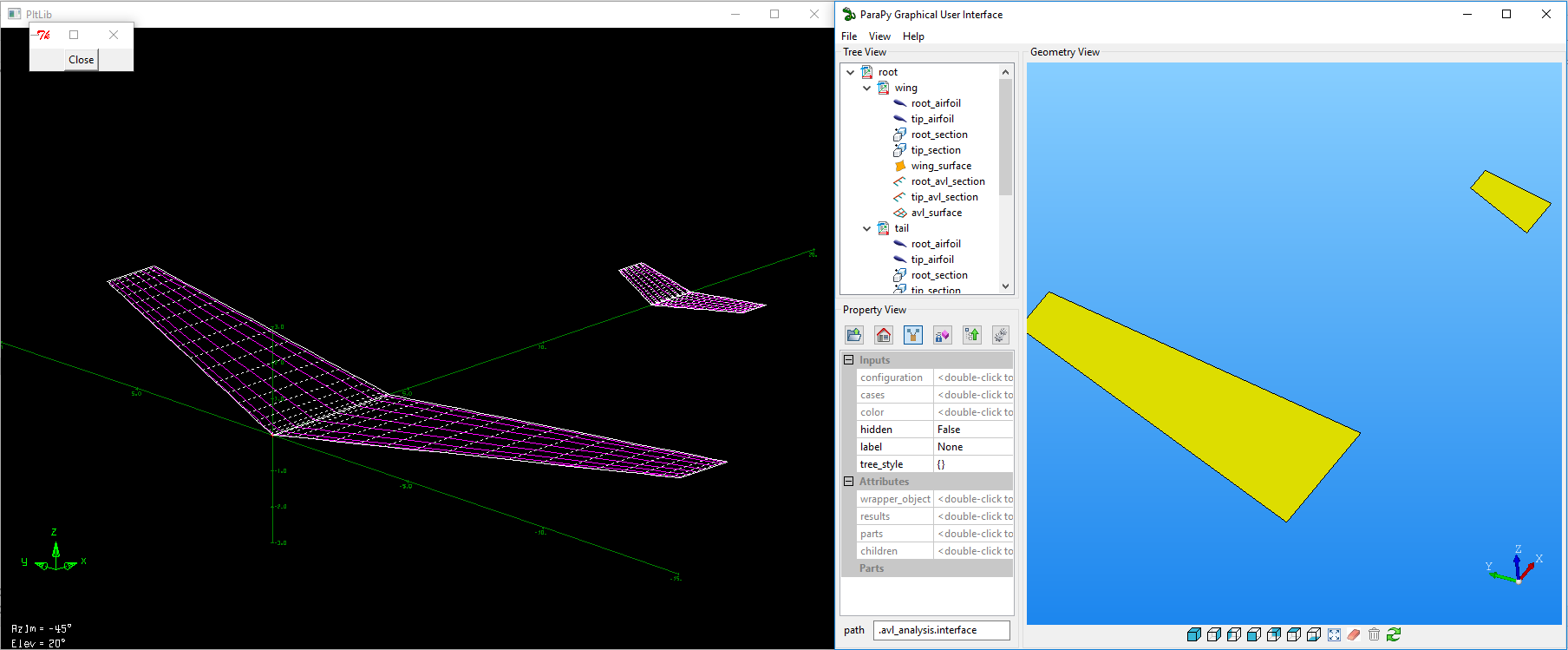
# AVL ParaPy module



AVL is a program for the aerodynamic and flight-dynamic analysis of rigid aircraft of arbitrary configuration. It employs an extended vortex lattice model for the lifting surfaces, together with a slender-body model for fuselages and nacelles. General nonlinear flight states can be specified. The flight dynamic analysis combines a full linearization of the aerodynamic model about any flight state, together with specified mass properties.

## Overview

The ParaPy AVL module provides ParaPy classes for the Python AVL wrapper.[[1]](#footnote-1) The current version supports definition of AVL geometry, cases, running AVL and parsing the results.

The classes provided by the module follow the structure of the AVL input file as described in the AVL documentation.[[2]](#footnote-2)

The main class is the “Interface”. By providing the Interface with an aircraft configuration and run cases, the analysis result can be found in the “results” attribute.

## Simple example

**from** \_\_future\_\_ **import** division  
**from** math **import** radians, tan  
**from** parapy.core **import** Base, Part, child, Attribute, Input  
**from** kbeutils.avl **import** (Interface, Case, Configuration, Section, NacaAirfoil,  
 Surface, Spacing, Control, Parameter)  
  
**class** AVLAnalysis(Base):  
  
 @Part  
 **def** aircraft(self):  
 **return** AircraftConfiguration(root\_chord=6.0,  
 taper\_ratio=0.4,  
 span=42,  
 sweep\_angle=24.0,  
 name=**"Example aircraft"**)  
  
 @Part  
 **def** interface(self):  
 **return** AVLInterface(configuration=self.aircraft,  
 case\_settings=[(**'fixed\_aoa'**, {**'alpha'**: 5}),  
 (**'flap\_cl'**,  
 {**'alpha'**: Parameter(name=**'alpha'**,  
 value=1.2,  
 constraint=**'CL'**),  
 **'flap'**: 15})])  
  
**class** AVLInterface(Interface):  
  
 case\_settings = Input()  
   
 *# overridden slot from Interface superclass* @Part  
 **def** cases(self):  
 **return** Case(quantify=len(self.case\_settings),  
 name=self.case\_settings[child.index][0],  
 settings=self.case\_settings[child.index][1])  
  
  
**class** AircraftConfiguration(Configuration):  
  
 root\_chord = Input()  
 taper\_ratio = Input()  
 span = Input()  
 sweep\_angle = Input()  
 naca\_airfoil = Input(**'2412'**)  
  
 flap\_hinge = Input(0.75)  
  
 @Part  
 **def** airfoil(self):  
 **return** NacaAirfoil(self.naca\_airfoil)  
  
 @Part  
 **def** root\_section(self):  
 **return** Section(chord=self.root\_chord,  
 airfoil=self.airfoil,  
 controls=[self.flap])  
  
 @Part  
 **def** tip\_section(self):  
 **return** Section(chord=self.root\_chord \* self.taper\_ratio,  
 airfoil=self.airfoil,  
 position=self.root\_section.position.translate(  
 **'x'**, self.span / 2 \* tan(radians(self.sweep\_angle)),  
 **'y'**, self.span / 2),  
 controls=[self.flap])  
  
 @Part  
 **def** flap(self):  
 **return** Control(name=**'flap'**,  
 gain=1,  
 x\_hinge=self.flap\_hinge,  
 dupplicate\_sign=1)  
  
 @Attribute  
 **def** mac(self):  
 t = self.taper\_ratio  
 **return** (2 / 3) \* self.root\_chord \* (1 + t + t \*\* 2) / (1 + t)  
  
 @Part  
 **def** wing\_surface(self):  
 **return** Surface(name=**"Wing"**,  
 n\_chordwise=8,  
 chord\_spacing=Spacing.cosine,  
 n\_spanwise=20,  
 span\_spacing=Spacing.cosine,  
 y\_duplicate=0.0,  
 sections=[self.root\_section, self.tip\_section])  
  
  
 *# overridden slots from Configuration superclass* @Attribute  
 **def** reference\_point(self):  
 **return** self.root\_section.position.translate(**'x'**, self.root\_chord / 4)  
  
 @Attribute  
 **def** reference\_chord(self):  
 **return** self.mac  
  
 @Attribute  
 **def** reference\_span(self):  
 **return** self.span  
  
 @Attribute  
 **def** reference\_area(self):  
 **return** self.wing\_surface.area \* 2  
  
 @Attribute  
 **def** surfaces(self):  
 **return** [self.wing\_surface]  
  
  
**if** \_\_name\_\_ == **'\_\_main\_\_'**:  
 **from** parapy.gui **import** display  
  
 analysis = AVLAnalysis()  
 display(analysis)

1. <https://github.com/renoelmendorp/AVLWrapper> [↑](#footnote-ref-1)
2. <http://web.mit.edu/drela/Public/web/avl/avl_doc.txt> [↑](#footnote-ref-2)