Preliminary Stability & Dynamics Study

Team 7:
Cooper LeComp, John A. Papas Dennerline,
Ian Greene, Christos Levy, Daniel Qi, Hans Suganda

26th September 2022

Contents

0.1 Preliminary Theoretical Basis

Based on inviscid analysis of an airfoil, there is a point in the airfoil called the aerodynamic center wherein the moments experienced about that point is insensitive to angle of attack. The concept of aerodynamic center can be extended to wings on an aircraft. The moment about the aerodynamic center of an aircraft wing is often non-zero, but it is unchanging regardless of the aircraft's orentiation.

0.1.1 Single Wing Thought Experiment

Let us consider a simple system where we have a single finite wing. Let us have some reference point P located about the zero-α line, but downwind of the wing.

The moment experienced about point P would be the moments about the aerodynamic center plus the influence of the lift force by the finite wing with the moment arm length evaluated at the aerodynamic center of the finite wing.

Imagine that this point P is located far away from the aerodynamic center of the finite wing. If the angle of attack is changed by say $5\,deg$, then this would result in a very large change of moments about point P because the moment arm of the changing lift force is large. As point P is moved closer and closer to the aerodynamic center of the finite wing, the change of $5\,deg$ in angle of attack affects the moments about point P less and less because the moment arm is increasingly small.

Eventually if point P is on the aerodynamic center of the finite wing, then the moment about P does not change with the angle of attack the wing is at.

0.1.2 Double Wing Thought Experiment

Let us now consider a slightly more complex system where we have 2 finite wings. Wing 1 is going to be upstream of wing 2 and wing 2 is going to be downstream of wing 1. Let us now have point P be located in between wing 1 and wing 2.

Now the moments about point P depends on the summation of the moments at the aerodynamic center of both wing 1 and 2 (unchanging with angle of attack) plus the influence of changing lift due to wing 1 and wing 2.

Suppose the angle of attack for the system was increased by $5\,deg$. Both wings 1 and 2 will provide additional lift and drag, but let us ignore the drag for now. The increase in lift of wing 1 would provide a moment that tends to pitch the system "up", in the clockwise direction. The increase in lift of wing 2 would provide a moment that tends to pitch the system "down", in the counter-clockwise direction.

If point P is moved backwards closer to wing 2, then the moments influence of wing 2 would be smaller when the angle of attack has been increased and the moments influence of wing 1 would be greater when the angle of attack is increased. This is because the moment arm length is smaller to wing 2 and greater for wing 1. Conversely, if we move point P front, closer to wing 1, then the moment influence of wing 1 is reduced in the event of a change of angle of attack due to smaller moment arm, and the influence of wing 2 is increased in the event of a change of angle of attack due to a larger moment arm.

Eventually, if the lift force increases linearly with angle of attack (valid assumption) there must be a unique point in between the 2 wings wherein the moments about that point is unchancing with angle of attack, because the influence of clockwise moments due to wing 1 is exactly cancelled by the influence of counter-clockwise moments due to wing 2. This is what is often known as the neutral point of the system.

0.1.3 Longitudinal Static Stability

It is often convenient to consider moments about the center of mass.

If the center of mass is behind the neutral point, the "influence" of wing 2 would be reduced due to a shortening of the moment arm, and the "influence" of wing 1 would be increased due to a lengthening of the moment arm. This means that if the 2-wing system (aircraft) pitches up slightly, wing 1 (forward wing) is going to have a winning influence compared to wing 2 (back wing) and the aircraft will have a tendency to keep pitching up. This means that an aircraft whose center of mass is behind the neutral point is statically unstable; a small pitch in any direction will give a larger tendency to deviate from level steady flight.

If the center of mass is in front of the neutral point, the "influence" of wing 1 would now be reduced due to the shortning of the moment arm, and the "influence" of wing 1 would be increased due to lengthening of the moment arm. This is obvoius because now the center of mass (the point we are evaluating moments at) is closer to wing 1 and further way from wing 2. This means now if the 2-wing aircraft system pitches up slightly, wing 2 (backward wing) is going to have a winning influence compared to wing 1 (front wing) and the aircraft would pitch back down. This means that for an aircraft to be statically stable, the center of mass has to be in front or upwind of the neutral point.

0.1.4 Trim Conditions

Arguably, the most important trim condition for an aircraft would be the longitudinal trim. We can model the lift produced by the finite wings to be a linear function of angle of attack. We can also model the drag produced by the finite wings as a quadratic function of lift. This means we can also model the drag produced by the finite wings as a quadratic function of angle of attack.

The lift to drag ratio is an incredibly important parameter in aerodynamic efficiency. Since the lift is equivalent to weight for level and steady flight, then the drag can be wholly determined from the lift to drag ratio for the entire aircraft. Remembering that lift is linear with angle of attack and drag is quadratic to angle of attack, a general expression of lift to drag ratio as a function of angle of attack,

$$\frac{L}{D} = \frac{k_1 + k_2\alpha}{k_3 + k_4\alpha + k_5\alpha^2}$$

We can find the optimum lift to drag ratio by differentiation,

$$\frac{d}{d\alpha}\left[\frac{L}{D}\right] = \frac{d}{d\alpha}\left[\frac{k_1 + k_2\alpha}{k_3 + k_4\alpha + k_5\alpha^2}\right]$$

$$\frac{d}{d\alpha}\left[\frac{L}{D}\right] = \frac{k_2(k_3 + k_4\alpha + k_5\alpha^2) - (k_1 + k_2\alpha)(k_4 + 2k_5\alpha)}{(k_3 + k_4\alpha + k_5\alpha^2)^2}$$

The expression above can be simplified algebraically, but if we set the derivative above to 0, we basically set the numerator of the expression above to 0. Since the numerator is an quadratic function of α, we have 2 maxima/minima points for the L/D. One is going to correspond to positive lift and the other to negative lift, so we can be assured that a single optimum angle of attack α would give us the best positive L/D. We will twist our entire wing to make sure that the aircraft flies at its optimum angle of attack at its ideal trim conditions.

0.2 Algorithm Development

0.2.1 Planning

1. From the Aerodynamic Preliminary Design, we should have an airfoil cross-section that suits our needs. We can design our wing from this airfoil-cross section

2. After we design our main wing, we need to determine its optimum angle of attack. We can run `avl` at a range of angles of attack and figure out the maximum Lift to Drag ratio achievable.

3. After we have our optimum angle of attack, we need to determine the moments of coefficient at our optimum angle of attack.

4. We then need an estimation of our center of mass. We will have to choose a static margin based on historical data. This static margin will control how "manuverable" our aircraft is going to be.

5. This permits a range of possibilities of tail boom and tail size. We will attempt to minimize: cost = weight of tail boom + weight of tail + downwash produced by tail at cruise.

6. We can make changes to the size of the rudder and adjust the dihedral angle to get rid of roll subsidence and dutch roll.

Longitudinal Modes:

1. Phugoid: Not important

2. Short Period Oscillations: Not that important

Lateral-Direction Modes:

1. Roll subsidence: This is affected by dihedral. If we have problems, just increase the dihedral.

2. Dutch Roll: Yaw Damper is supposed to get rid of this, increase size of tail if necessary

3. Spiral Divergence: Deadly and IDK how to offset

0.2.2 Trim_Wing

This is a `Bash` script which calls on all of the other functionalities to perform an aerodynamic "trim" of the wing.

```
#!/bin/bash

#Generate the Wing Configuration File
./Gen_Wing_AVL.py Wing_Parameters.tex Naca_6412.dat > Wing_Trim/Simple_Taper.avl

#Generate avl Instructions
./Gen_Instr Wing_Trim/ 5 0.3 24

#Run avl
cd Wing_Trim/
avl Simple_Taper.avl < Stab_Instr_AVL.txt

#Move the data out
mv Outhouse.txt ..

#Get out of there
cd ..

#Cleanup the data for Python
./Clean_Data Outhouse.txt

#Figure out final trim angle
./Comp_Trim_Angle.py Trim_Main_Wing.tex
```

0.2.3 Gen_Wing_AVL.py

The solver of choice is `avl` due to its scriptability. We need to produce a text file that contains a textual description of the "aircraft" we want to model. Because we are "trimming" the main wing of the aircraft, we this `Python` script only defines the bare minimum for a wing. Note the wing type which is a tapered wing desribed in this script is essentially chosen from the Aerodynamics Preliminary Design and just because of ease of manufacturing. Note that this takes an input file from the previous aerodynamic study.

```
#!/bin/python

#Library import statements
import numpy as np
import sys

#This is the name of the specified input file
Input_File = str(sys.argv[1])
#This is what choice of airfoil to use
Airfoil_Selection = str(sys.argv[2])

#Reading in the data
data = np.genfromtxt(Input_File)

#Writing for incompressible FLows
print("Simple_Tapered_Wing")
print("#Mach\n", 0.0)
print("#IYsym␣␣␣␣IZsym␣␣␣␣Zsym\n", 0, 0, 0)
print("#Sref␣␣␣␣␣Cref␣␣␣␣␣Bref")

#WRiting Reference area, chord reference and span reference
print(data[0], end="␣") #Writing of Reference Area
print(data[2], end="␣") #Writing of Reference Chord
print(data[4]) #Writing of Reference Wing Span

#TODO:AUTOMATE THIS PART
#Writing Center of Mass (Unimportant)
print("#Xref␣␣␣␣␣Yref␣␣␣␣␣Zref\n", 0.3023*data[2],0,0)

#TODO:AUTOMATE THIS PART
#Writing Drag obtained from XFoil
print("#CDp\n", 0.05597)

#Surface Header
print("#############################################################################")
print("SURFACE")
print("WING")
print("#Nchordwise␣␣␣␣␣␣Cspace␣␣Nspanwise␣␣␣Sspace")
print(30, 1.0, 30, 1.0)
print("YDUPLICATE\n", 0.0)
print("ANGLE\n", 0.0)
print("#␣x,y,z␣bias␣for␣whole␣surface")
print("TRANSLATE\n", 0.0, 0.0, 0.0)

#Definition of wing at the roots
print("#############################################################################")
print("SECTION")
print("#␣␣␣␣Xle␣␣␣␣␣␣Yle␣␣␣␣␣␣Zle␣␣␣␣␣␣chord␣␣␣␣␣␣␣␣angle␣␣␣␣␣␣␣␣Nspan␣␣␣␣␣␣␣␣Sspace")
print(0, 0, 0, data[2], 0, 30, 1.0)
print("AFIL\n", Airfoil_Selection)

#Definition of wing at the tips
print("#############################################################################")
print("SECTION")
print("#␣␣␣␣Xle␣␣␣␣␣␣Yle␣␣␣␣␣␣Zle␣␣␣␣␣␣chord␣␣␣␣␣␣␣␣angle␣␣␣␣␣␣␣␣Nspan␣␣␣␣␣␣␣␣Sspace")
```

```
print((data[2]-data[3])/2, data[4]/2, 0, data[3], 0, 30, 1.0)
print("AFIL\n", Airfoil_Selection)
```

To use this `Python` script, invoke the command,

```
python Gen_Wing_AVL.py Wing_Parameters.tex Naca_6412.dat
```

The contents of the file `Wing_Parameters.tex` is shown below,

```
#Wing Area (m^2)|Taper Ratio|Wing Root Chord (m)|Tip Root Chord (m)|Wing Span (m)|Aspect Ratio
0.75 0.45 0.45 0.2025 2.2988505747126435 7.046285286475535
```

A sample output of this is shown below,

```
Simple_Tapered_Wing
#Mach
 0.0
#IYsym   IZsym   Zsym
 0 0 0
#Sref    Cref    Bref
0.75 0.45 2.2988505747126435
#Xref    Yref    Zref
 0.13603500000000002 0 0
#CDp
 0.05597
#################################################################
SURFACE
WING
#Nchordwise     Cspace  Nspanwise   Sspace
30 1.0 30 1.0
YDUPLICATE
 0.0
ANGLE
 0.0
# x,y,z bias for whole surface
TRANSLATE
 0.0 0.0 0.0
#################################################################
SECTION
#   Xle     Yle     Zle     chord       angle       Nspan       Sspace
0 0 0 0.45 0 30 1.0
AFIL
 Naca_6412.dat
#################################################################
SECTION
#   Xle     Yle     Zle     chord       angle       Nspan       Sspace
0.12375 1.1494252873563218 0 0.2025 0 30 1.0
AFIL
 Naca_6412.dat
```

0.2.4 Gen_Instr

This `Bash` script generates the instruction for `avl` to make and run all of the test cases. It's outputs are rather complicated especially if you want a large number of increments,

```
#!/bin/bash

#Arg1: Name of address to write Instructions to
#Arg2: Starting Angle of Attack
#Arg3: Increments of AOA's
#Arg4: Number of AOA increments

#Basic Names of the instruction file
Base_Instr_Name="Stab_Instr_AVL.txt"
```

```
#Name of AVL Output File
Out_File="Outhouse.txt"

#Appending the address of where instruction files should be written to
Full_Instr_Name=$1$Base_Instr_Name

#Enter Operational mode
echo "OPER" > $Full_Instr_Name

#Iterate over the different increments
for ((index = 1; index <= $4; index=index+1)) ;do
    #Set the Angle of Attack
    AOA=$(echo $2"+"$index"*"$3 | bc)
    echo "A␣A␣" $AOA >> $Full_Instr_Name
    #Execute the Case
    echo "X" >> $Full_Instr_Name
    #Write into File
    echo "W" >> $Full_Instr_Name
    if [ $index -eq 1 ]; then
        #For the first run case, enter the output filename
        echo $Out_File >> $Full_Instr_Name
    else
        #For subsequent cases, just leave it blank, avl should know
        echo "␣" >> $Full_Instr_Name
    fi
    #Create New Case
    echo + >> $Full_Instr_Name
done

#Exit Operation Mode
echo "␣" >> $Full_Instr_Name

#Quit AVL
echo "QUIT" >> $Full_Instr_Name
```

A small part of the output is shown below,

```
OPER
A A  5.3
X
W
Outhouse.txt
+
A A  5.6
X
W

+
...
```

0.2.5 Clean_Data

So the text file named `Outhouse.txt` (appropriately named) produced by `avl` is in a messy format. The `Bash` script here basically extracts only the important useful things and compiles them into a file which can be conveniently read in `Python`.

```
#!/bin/bash

#Arg1: Name of Data File

#Clean Up the alpha file
Alpha_File="ALPHA_DELETE_THIS.txt"
```

```
grep 'Alpha' $1 > $Alpha_File
sed -i 's/Alpha =//' $Alpha_File
sed -i 's/p/\n/' $Alpha_File
sed -i '/^b/d' $Alpha_File

#Clean up the total Lift coefficeint File
CL_tot_File="CL_DELETE_THIS.txt"
grep 'CLtot' $1 > $CL_tot_File
sed -i 's/CLtot =//' $CL_tot_File

#Clean up the Total Drag Coefficient File
CD_tot_File="CD_DELETE_THIS.txt"
grep 'CDtot' $1 > $CD_tot_File
sed -i 's/CDtot =//' $CD_tot_File

#Re-arrange into columns
paste $Alpha_File $CL_tot_File $CD_tot_File | column -s $'\t' -t > Trim_Main_Wing.tex

#Clean-up intermediate files
rm $Alpha_File $CL_tot_File $CD_tot_File

#Add the file headers
sed -i '1i\#Angle of Attack (deg) | Lift Coefficient | Drag Coefficient' Trim_Main_Wing.tex
```

0.2.6 Comp_Trim_Angle.py

This `Python` file takes in the already cleaned data from `Outhouse.txt` and figures out the maximum lift to drag ratio as well as the correspoonding angle of attack.

```
#!/bin/python

#Library Import Statements
import numpy as np
import sys

#Name of Data File
Dat_File = str(sys.argv[1])

#Read in Data File
data = np.genfromtxt(Dat_File)

#Comutation of Lift to Drag Ratio
LDR = data[:,1]/data[:,2]

#Location of Maximum Lift to Drag Ratio
LDR_M_Loc = np.argmax(LDR)

#Maximum Lift to Drag Ratio
LDR_Max = LDR[LDR_M_Loc]

#Angle of Attack which Max L/D Occurs
alpha_LDR_Max = data[LDR_M_Loc, 0]

print("#Alpha L/D Max (deg) | L/D max Value")
print(alpha_LDR_Max, LDR_Max)
```

0.2.7 Plot_Wing_Performance.py

This `Python` script is only used for data visualization and to plot a predicted lift curve slope compared to the `avl` results. This is used to generate a "better" view of the whole story.

```
#!/bin/python

#Library Import Statements
import numpy as np
import matplotlib.pyplot as plt
import sys

#Name of Data File
Dat_File = str(sys.argv[1])

#Read in Data File
data = np.genfromtxt(Dat_File)
A_r = 7.046285286475535
zero_loc = np.argmin(data[:,0]**2)
#This is the alpha values for the predicted line
alpha_pred = [data[zero_loc,0], data[-1,0]]
#This is the approximate derivative of the predicted lift curve
cl_grad_pred = 2*np.pi*(A_r/(A_r+2))
#This is predicted lift coeff at lower value of alpha, make it match
cl_lower = data[zero_loc, 1]
#This is higher predicted lift coeff at higher value of alpha
cl_upper = cl_lower + cl_grad_pred*np.deg2rad(alpha_pred[1]-alpha_pred[0])
#Forming the cl predicted array
cl_pred = [cl_lower, cl_upper]

#Plot Lift Coefficient Vs Alpha
figure_CL, axis_CL = plt.subplots()
axis_CL.plot(data[:,0], data[:,1], 'kx-',label='AVL Wing')
axis_CL.plot(alpha_pred, cl_pred, 'r--',label='Predicted')
axis_CL.set_xlabel('Angle of Atack')
axis_CL.set_ylabel('Lift Coefficient')
axis_CL.set_title('Lift Curve of a Wing')
axis_CL.legend()
axis_CL.grid()

#Plot Drag Curve
figure_CD, axis_CD = plt.subplots()
axis_CD.plot(data[:,0], data[:,2], 'rx-',label='Wing')
axis_CD.set_xlabel('Angle of Atack')
axis_CD.set_ylabel('Drag Coefficient')
axis_CD.set_title('Drag Curve of a Wing')
axis_CD.legend()
axis_CD.grid()

#Plot Drag Polar
figure_CDCL, axis_CDCL = plt.subplots()
axis_CDCL.plot(data[:,1], data[:,2], 'gx-',label='Wing')
axis_CDCL.set_xlabel('Lift Coefficient')
axis_CDCL.set_ylabel('Drag Coefficient')
axis_CDCL.set_title('Drag Polar of a Wing')
axis_CDCL.legend()
axis_CDCL.grid()

#Plot L/D Ratio
figure_LDR, axis_LDR = plt.subplots()
axis_LDR.plot(data[:,0], data[:,1]/data[:,2], 'cx-',label='Wing')
axis_LDR.set_xlabel('Angle of Attack (deg)')
axis_LDR.set_ylabel('L/D Ratio')
axis_LDR.set_title('L/D Ratio for a wing')
axis_LDR.legend()
axis_LDR.grid()

#Show Plots
```

```
plt.show()
```

0.3 Results

The results for the lift to drag ratio seems unrealistic and so is the angle of attack. The main issue is that `avl` is a vortex lattice method, which is an inviscid solver. This means that drag due to viscous effects are severely underestimated. Since drag is the denominator in the lift to drag ratio, this inaccurate predictions of drag leads to wildly inaccurate results for the main wing's optimum angle of attack.

We have also attempted to "inject" viscous drag taken from `xfoil` directly for the airfoil operating at 50000 Reynold's number and the results are still unrealistic. On the other hand, the lift curve slope matches well with the approximate expression, indiciating that the approximation for the lift curve slope is usable for modelling purposes.