File - C:\Users\geoff\Desktop\FlyOx Concept\Python\engines\wingEngine.py

```
__author__ = 'Geoffrey Nyaga'

import sys
sys.path.append('../')
from API.db_API import write_to_db,read_from_db

from math import sqrt,cos,sin,pi,log,tan
import numpy as np
import matplotlib.pyplot as plt

import API.wingAPI as wapi

S = read_from_db('S') *10.76 #sq ft
taper = read_from_db('taper')


#IMPORT SOME OF THE VALUES HERE
initialWeight = read_from_db('initialWeight')
finalWeight = read_from_db('finalWeight')
finalMTOW = read_from_db('finalMTOW')
altitude = read_from_db('cruise_altitude') #ft
rhoSL = read_from_db('rhoSL')


cbhp = read_from_db('cbhp')

propEff = read_from_db('propEff')
cruiseSpeed =read_from_db('maxSpeed')/1.2 #CALCULATE THIS, MAYBE USE MAXSPEED/1.2 THEN LATER OPTIMIZE IN PERFORMANCE
Range = read_from_db('Range')*3280.84#ft

cdMin = 0.02541   #CALCULATE THIS SOMEHOW CZ WE HAVE IMPORTED IT INTO THE VALUES FILES
endurance = 5.172 #hours

### check spelling from previous modules
ldMax = read_from_db('ldMax')
rhoSL = read_from_db('rhoSL')
altitudeDensity = read_from_db('altitudeDensity')



write_to_db('cdMin',cdMin)
write_to_db('cbhp',cbhp)
write_to_db('cruiseSpeed',cruiseSpeed)



wing = wapi.aspectRatio(initialWeight,finalWeight,cruiseSpeed,S,rhoSL,altitude,cbhp,propEff,Range,cdMin,endurance,ldMax)

# print(wing.altitudeDensity(),"Altitude Density")
# print(wing.cruiseCL(),"Cruise CL")
# print (wing.ct(),"ct")
# print (wing.rangeAR(),"Range AR")
# print (wing.enduranceAR(),"Endurance AR")
# print ('For Sailplanes')
# print (wing.unPoweredSailplaneAR(),"unpowered Sailplane AR")
# print (wing.poweredSailplaneAR(),"powered Sailplane AR")

altitudeDensity = (wing.altitudeDensity())
cruiseCL = (wing.cruiseCL())
ct= wing.ct()
rangeAR =  (wing.rangeAR())
enduranceAR=  (wing.enduranceAR())
unPoweredSailplaneAR = (wing.unPoweredSailplaneAR())
poweredSailplaneAR =(wing.poweredSailplaneAR())

write_to_db('altitudeDensity',altitudeDensity)
write_to_db('cruiseCL',cruiseCL)
write_to_db('ct',ct)
write_to_db('rangeAR',rangeAR)
write_to_db('enduranceAR',enduranceAR)
write_to_db('unPoweredSailplaneAR',unPoweredSailplaneAR)
write_to_db('poweredSailplaneAR',poweredSailplaneAR)


wing1 = wapi.wingDimensions(S,wing.rangeAR(),taper)
```

```
 76
 77
 78
 79 wingSpan =  (wing1.wingSpan())
 80 averageChord =  (wing1.Cavg())
 81 rootChord =  (wing1.rootChord())
 82 tipChord =  (wing1.tipChord())
 83 meanGeometricChord =  (wing1.meanGeometricChord())
 84 chordAtY =  (wing1.chordAtY(10))   #CHANGE 10 LATER TO INPUT
 85 # chordAtY =  (wing1.chordAtY(float(input('enter lateral distance for chordAtY:'))))
 86
 87 yMGC = (wing1.yMGC())
 88 write_to_db('wingSpan',wingSpan)
 89 write_to_db('averageChord',averageChord)
 90 write_to_db('rootChord',rootChord)
 91 write_to_db('tipChord',tipChord)
 92 write_to_db('meanGeometricChord',meanGeometricChord)
 93 write_to_db('chordAtY',chordAtY)
 94 write_to_db('yMGC',yMGC)
 95
 96 ### Gudmundsson says that taper ratio is the second most important geometric properties of the wing
 97
 98 ## ldMax as a funtion of Aspect ratio
 99 ARrange = np.arange(10,36)
100 ldmaxPowered = (1.7405*ARrange) - 0.443
101 ldmaxunpowered = (0.0352*ARrange**2 )+ ((3.1315*ARrange) - 10.787)
102
103 plt.plot(ARrange,ldmaxPowered,label = "Powered")
104 plt.plot(ARrange,ldmaxunpowered,label = "Unpowered")
105 plt.xlabel('Wing Aspect Ratio')
106 plt.ylabel('Maximum Lift-to-Drag Ratio')
107 plt.title(" SAILPLANES \n ldMax as a funtion of Aspect ratio")
108 plt.legend()
109 if __name__ == '__main__':
110     plt.show()
111
112 #IMPORT ALL THESE JUNK SOMETIMES LATER... AND JEEZ ORGANIZE THE DAMN CODE
113 ## 2D airfoil properties
114 ## this will kill you down the line Nyaga
115 AOA = 5 #Degree
116 clalfa = 6.1
117 clo = 0.4
118 alfazero = -clo/(clalfa/57.3)   #confirm this or just read the damn graph
119 cma = -0.01
120 clmax = 1.560  #from airfoil
121 #do shit here later e.g Re on tip and root is different cz of taper. Read from the NACA R-824
122 clmaxRoot = 1.561
123 clmaxTip = 1.4
124
125 write_to_db('AOA',AOA)
126 write_to_db('clalfa',clalfa)
127 write_to_db('clo',clo)
128 write_to_db('alfazero',alfazero)
129 write_to_db('cma',cma)
130 write_to_db('clmax',clmax)
131 write_to_db('clmaxRoot',clmaxRoot)
132 write_to_db('clmaxTip',clmaxTip)
133
134 angularStation = np.array([22.5,45.0,67.5,90.0])/57.296
135 CosangularStation = np.cos (angularStation)
136 cTheta = wing1.tipChord()*CosangularStation + ( wing1.rootChord()*(1-CosangularStation))
137 SinangularStation = np.sin(angularStation)
138 SinangularStation3 = np.sin(angularStation*3)
139 SinangularStation5 = np.sin(angularStation*5)
140 SinangularStation7 = np.sin(angularStation*7)
141 angularMu = cTheta*clalfa/(4*wing1.wingSpan())
142
143 A11 = SinangularStation*(angularMu + SinangularStation)
144 A12 = SinangularStation3*(3*angularMu + SinangularStation)
145 A13 = SinangularStation5*(5*angularMu + SinangularStation)
146 A14 = SinangularStation7*(7*angularMu + SinangularStation)
147 A = np.array([A11,A12,A13,A14])
148 Afinal = A.transpose()
149
150 B1 = angularMu*(AOA/57.296 - alfazero/57.296)*SinangularStation
151 Bfinal = B1.reshape(4,1)
152
```

File - C:\Users\geoff\Desktop\FlyOx Concept\Python\engines\wingEngine.py

```
153 #c = Afinal**-1   # THIS METHOD DOESNT WORK FOR MATRIX INVERSE...SH*T AINT LIKE MATLAB HAHA 1 HOUR
    WASTED
154 c = np.linalg.inv(Afinal)
155 d = np.dot(c,Bfinal)
156 d0 = d[0]
157
158
159 cl = np.pi*wing.rangeAR()*d0
160 # print (cl,"cl")
161
162 dFactor = 3*(d[1]/d[0])**2 + (5*(d[2]/d[0])**2) + (7*(d[3]/d[0])**2)
163
164 CDi = cl**2*(1+dFactor)/(np.pi*wing.rangeAR())
165 # print(CDi,"CDi")
166 reducedCDi = CDi[0]
167 write_to_db('reducedCDi',reducedCDi)
168
169 reducedOswaldEff = 1/(1+dFactor)
170 # print(reducedOswaldEff,"oswaldEff")
171 reducedOswaldEff = reducedOswaldEff[0]
172 write_to_db('reducedOswaldEff',reducedOswaldEff)
173
174
175 CLalfa = (cl/((AOA/57.296) - (alfazero/57.296)))
176 CLalfa = CLalfa[0]
177 # print (CLalfa , "per radian")
178 write_to_db('CLalfa',CLalfa)
179
180 reducedMaxSpeed = sqrt ((2*finalMTOW)/(rhoSL*S*cl))
181
182 write_to_db('reducedMaxSpeed',reducedMaxSpeed)
183 # print (vel,"ft/s")
184
185
186 #Accounting for the fuselage
187 fuselageWidth = 4.167 #ft
188 write_to_db('fuselageWidth',fuselageWidth)
189
190 reducedWingspan = wing1.wingSpan() - fuselageWidth
191 reducedS = S - (wing1.rootChord()*fuselageWidth)
192 reducedAR = (reducedWingspan**2)/reducedS
193 reducedRootChord= (wing1.rootChord()*(1-(fuselageWidth/wing1.wingSpan())) )+(wing1.tipChord()*(
    fuselageWidth/wing1.wingSpan()))
194 reducedTaper = (wing1.wingSpan()*wing1.rootChord())/(wing1.rootChord()*(wing1.wingSpan()-
    fuselageWidth)+(fuselageWidth*wing1.rootChord()))
195
196
197 # print ("CL of the reduced Wing")
198 reducedCL = ((2*finalMTOW)/(rhoSL*reducedMaxSpeed**2*reducedS)) #it actually increased. The name
    reducedCl is just for formality
199 write_to_db('reducedCL',reducedCL)
200
201 # print(reducedCL,"CL due to reduced Wing")
202 # print(reducedWingspan,"ft, reduced wingspan")
203 # print(reducedS,"ft^2, reduced Wing Area")
204 # print(reducedAR,"reduced AR" )
205 # print(reducedRootChord,"reduced Root Chord" )
206 # print(reducedTaper,"reduced Taper" )
207
208 #for SOME VERY WIERD REASON WE'LL GET INACCURATE DICT VALUES IF ANY OF THE FOLLOWING IS THE SAME WITH
     ANOTHER VALUE
209 sweepHalfChord = 4
210 sweepQuarterChord = 4.0
211 sweepLeadingEdge = 0
212 sweepTmax = 4.5
213
214 write_to_db('sweepHalfChord',sweepHalfChord)
215 write_to_db('sweepQuarterChord',sweepQuarterChord)
216 write_to_db('sweepLeadingEdge',sweepLeadingEdge)
217 write_to_db('sweepTmax',sweepTmax)
218
219
220 wing4 = wapi.classOswaldEff(rangeAR , sweepLeadingEdge, sweepTmax,fuselageWidth,wingSpan,cdMin)
221 if taper == 1:
222     oswaldEff = wing4.straightWingOswaldEff()
223     print (oswaldEff,"final oswald efficiency of this straight wing")
224 else:
```

File - C:\Users\geoff\Desktop\FlyOx Concept\Python\engines\wingEngine.py

```
225     oswaldEff = (wing4.sweptWingOswaldEff() + wing4.brandtOswaldEff() + wing4.douglasOswalfEff()) / 3
226     print (oswaldEff,"final oswald")
227 
228 write_to_db('oswaldEff',oswaldEff)
229 
230 # print(wing4.douglasOswalfEff(),'final oswaldEff')
```