```python
# coding: utf-8
__author__ = 'Geoffrey Nyaga'


import numpy as np
import math
import matplotlib.pylab as plt

def llt(N,S,AR,taper,alpha_twist,i_w,a_2d,alpha_0):

    b = math.sqrt(AR*S) # wing span (m)
    MAC = S/b # Mean Aerodynamic Chord (m)
    Croot = (1.5*(1+taper)*MAC)/(1+taper+taper**2) # root chord (m)
    # theta = np.arange(math.pi/(2*N), math.pi/2, math.pi/(2*(N)))
    theta = np.linspace((math.pi/(2*N)), (math.pi/2),N,endpoint=True)
    # alpha =np.arange(i_w+alpha_twist,i_w ,-alpha_twist/(N))
    alpha = np.linspace(i_w+alpha_twist,i_w ,N)
    z = (b/2)*np.cos(theta)
    c = Croot * (1 - (1-taper)* np.cos(theta)) # Mean Aerodynamics
    mu = c * a_2d / (4 * b)

    LHS = (mu * (np.array(alpha)-alpha_0)/57.3)#.reshape((N-1),1)# Left Hand Side

    RHS = []
    for i in range(1,2*N+1,2):
      RHS_iter = (np.sin(i*theta)*(1+(mu*i)/(np.sin(list(theta)))))#.reshape(1,N)
      # print(RHS_iter,"RHS_iter shape")
      RHS.append(RHS_iter)

    test = (np.asarray(RHS))
    x = np.transpose(test)
    inv_RHS = np.linalg.inv(x)
    ans = np.matmul(inv_RHS,LHS)

    mynum = np.divide((4*b),c)
    test = ((np.sin((1)*theta))*ans[0]*mynum)
    test1 = ((np.sin((3)*theta))*ans[1]*mynum)
    test2 = ((np.sin((5)*theta))*ans[2]*mynum)
    test3 = ((np.sin((7)*theta))*ans[3]*mynum)
    test4 = ((np.sin((9)*theta))*ans[4]*mynum)
    test5= ((np.sin((11)*theta))*ans[5]*mynum)
    test6 = ((np.sin((13)*theta))*ans[6]*mynum)
    test7 = ((np.sin((15)*theta))*ans[7]*mynum)
    test8 = ((np.sin((17)*theta))*ans[8]*mynum)

    CL = test+test1+test2+test3+test4+test5+test6+test7+test8
    CL1 = np.append (0,CL)
    y_s=[b/2 , z[0], z[1], z[2] ,z[3], z[4] ,z[5], z[6], z[7] ,z[8]]
    # print(y_s,"ys")

    if __name__ == "__main__":
        plt.plot(y_s,CL1, marker='o')
        plt.title('Lifting Line Theory\n Elliptical Lift distribution')
        plt.xlabel('Semi-span location (m)')
        plt.ylabel ('Lift coefficient')
        plt.grid()
        if __name__ == "__main__":
            plt.show()


    CL_wing = math.pi * AR * ans[0] # USE THIS CL WITH CRUISE SPEED TO CALCULATE THE ACCURATE LIFT !!!!!!!!!!
    myfinalmat = np.array(y_s)
    myfinalmat2 = CL1

    return myfinalmat,myfinalmat2, CL_wing #CL_wing,y_s,CL1

    if __name__ == "__main__":
        print (CL_wing,"CL_wing")

def llt_with_plots(N,S,AR,taper,alpha_twist,i_w,a_2d,alpha_0):

    b = math.sqrt(AR*S) # wing span (m)
    MAC = S/b # Mean Aerodynamic Chord (m)
    Croot = (1.5*(1+taper)*MAC)/(1+taper+taper**2) # root chord (m)
    # theta = np.arange(math.pi/(2*N), math.pi/2, math.pi/(2*(N)))
    theta = np.linspace((math.pi/(2*N)), (math.pi/2),N,endpoint=True)
```

```python
    # alpha =np.arange(i_w+alpha_twist,i_w ,-alpha_twist/(N))
    alpha = np.linspace(i_w+alpha_twist,i_w ,N)
    z = (b/2)*np.cos(theta)
    c = Croot * (1 - (1-taper)* np.cos(theta)) # Mean Aerodynamics
    mu = c * a_2d / (4 * b)

    LHS = (mu * (np.array(alpha)-alpha_0)/57.3)#.reshape((N-1),1)# Left Hand Side

    RHS = []
    for i in range(1,2*N+1,2):
      RHS_iter = (np.sin(i*theta)*(1+(mu*i)/(np.sin(list(theta)))))#.reshape(1,N)
      # print(RHS_iter,"RHS_iter shape")
      RHS.append(RHS_iter)

    test = (np.asarray(RHS))
    x = np.transpose(test)
    inv_RHS = np.linalg.inv(x)
    ans = np.matmul(inv_RHS,LHS)

    mynum = np.divide((4*b),c)
    test = ((np.sin((1)*theta))*ans[0]*mynum)
    test1 = ((np.sin((3)*theta))*ans[1]*mynum)
    test2 = ((np.sin((5)*theta))*ans[2]*mynum)
    test3 = ((np.sin((7)*theta))*ans[3]*mynum)
    test4 = ((np.sin((9)*theta))*ans[4]*mynum)
    test5= ((np.sin((11)*theta))*ans[5]*mynum)
    test6 = ((np.sin((13)*theta))*ans[6]*mynum)
    test7 = ((np.sin((15)*theta))*ans[7]*mynum)
    test8 = ((np.sin((17)*theta))*ans[8]*mynum)

    CL = test+test1+test2+test3+test4+test5+test6+test7+test8
    CL1 = np.append (0,CL)
    y_s=[b/2 , z[0], z[1], z[2] ,z[3], z[4] ,z[5], z[6], z[7] ,z[8]]

    # num1 =    str(round(alpha_twist,2))
    num1 = "{0:.2f}".format(alpha_twist)
    num2 = "{0:.2f}".format(i_w)

    plt.plot(y_s,CL1, marker='o',label = ( "alpha_twist:",num1,"wing_incidence:",num2   )  )
    plt.title('Lifting Line Theory\n Elliptical Lift distribution')
    plt.xlabel('Semi-span location (m)')
    plt.ylabel ('Lift coefficient')
    plt.legend ()
    plt.grid()
    # plt.show()

    CL_wing = math.pi * AR * ans[0] # USE THIS CL WITH CRUISE SPEED TO CALCULATE THE ACCURATE LIFT !!!!!!!!!!
    return CL_wing

    if __name__ == "__main__":
        print (CL_wing,"CL_wing")


def llt_subplots(N,S,AR,taper,alpha_twist,i_w,a_2d,alpha_0):

    b = math.sqrt(AR*S) # wing span (m)
    MAC = S/b # Mean Aerodynamic Chord(m)
    Croot = (1.5*(1+taper)*MAC)/(1+taper+taper**2) # root chord (m)
    # theta = np.arange(math.pi/(2*N), math.pi/2, math.pi/(2*(N)))
    theta = np.linspace((math.pi/(2*N)), (math.pi/2),N,endpoint=True)
    # alpha =np.arange(i_w+alpha_twist,i_w ,-alpha_twist/(N))
    alpha = np.linspace(i_w+alpha_twist,i_w ,N)
    z = (b/2)*np.cos(theta)
    c = Croot * (1 - (1-taper)* np.cos(theta)) # Mean Aerodynamics
    mu = c * a_2d / (4 * b)

    LHS = (mu * (np.array(alpha)-alpha_0)/57.3)#.reshape((N-1),1)# Left Hand Side

    RHS = []
    for i in range(1,2*N+1,2):
      RHS_iter = (np.sin(i*theta)*(1+(mu*i)/(np.sin(list(theta)))))#.reshape(1,N)
      # print(RHS_iter,"RHS_iter shape")
      RHS.append(RHS_iter)

    test = (np.asarray(RHS))
    x = np.transpose(test)
```

```python
    inv_RHS = np.linalg.inv(x)
    ans = np.matmul(inv_RHS,LHS)

    mynum = np.divide((4*b),c)
    test = ((np.sin((1)*theta))*ans[0]*mynum)
    test1 = ((np.sin((3)*theta))*ans[1]*mynum)
    test2 = ((np.sin((5)*theta))*ans[2]*mynum)
    test3 = ((np.sin((7)*theta))*ans[3]*mynum)
    test4 = ((np.sin((9)*theta))*ans[4]*mynum)
    test5= ((np.sin((11)*theta))*ans[5]*mynum)
    test6 = ((np.sin((13)*theta))*ans[6]*mynum)
    test7 = ((np.sin((15)*theta))*ans[7]*mynum)
    test8 = ((np.sin((17)*theta))*ans[8]*mynum)

    CL = test+test1+test2+test3+test4+test5+test6+test7+test8
    CL1 = np.append (0,CL)
    y_s=[b/2 , z[0], z[1], z[2] ,z[3], z[4] ,z[5], z[6], z[7] ,z[8]]

    # num1 =    str(round(alpha_twist,2))
    num1 = "{0:.2f}".format(alpha_twist)
    num2 = "{0:.2f}".format(i_w)

    CL_wing = math.pi * AR * ans[0] # USE THIS CL WITH CRUISE SPEED TO CALCULATE THE ACCURATE LIFT !!!!!!!!!!
    return CL_wing

    if __name__ == "__main__":
        print (CL_wing,"CL_wing")

```