ASEN 3111 - CFD Lab - Main

Table of Contents

```
Consolodates the data from the CFD simulations and compares the results
to thin airfoil and vortex panel method.

Author: Samuel Razumovskiy
Date: 12/12/2019
```

(Knock knock) House keeping

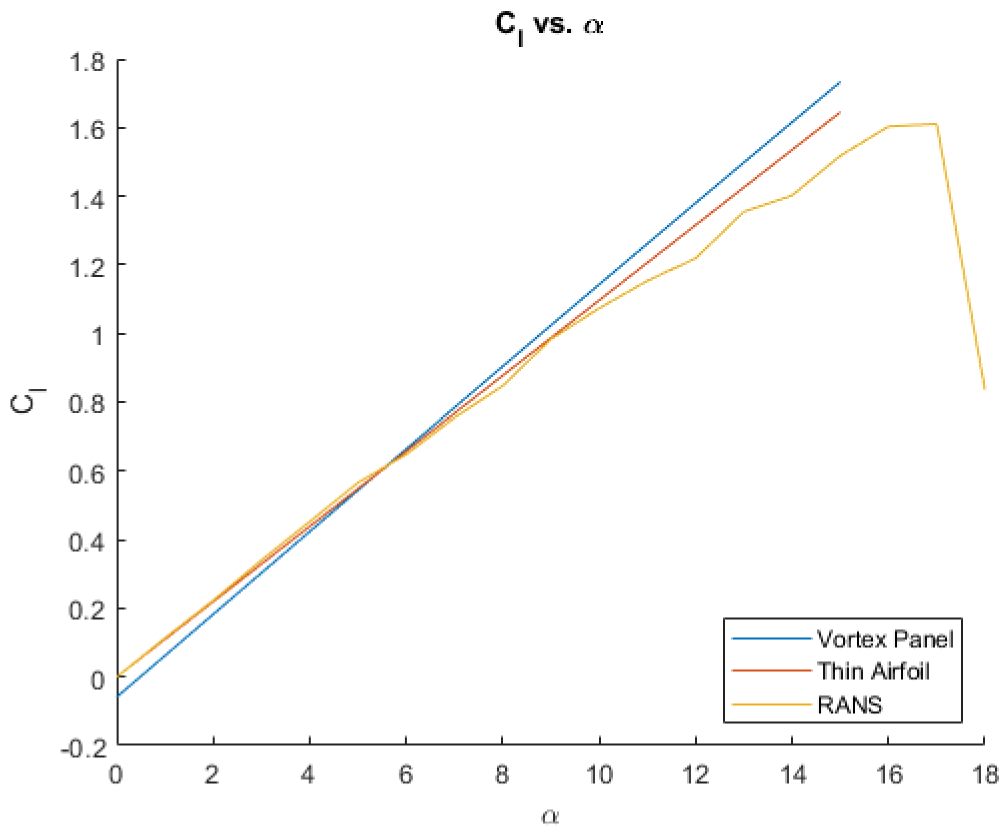
```
clear,clc,close all
```

Problem 1

```
T = readtable('Cl and Cd.xlsx','Range','D:E');

m = 0;
p = 0;
t = 12;
c = 1;
N = 150;
M = N-2;
v_inf = 52.06373;
[x,y,yc] = NACA_Airfoils(m,p,t,c,N);
alphas = linspace(0,15);
for j = 1:numel(alphas)
    alpha = alphas(j)*pi/180;
    [cl(j)] = Vortex_Panel(alpha,v_inf,c,x,y,M);
end

% Calculating the dz_dx
dz_dx = 0;
% Calculating theta
theta = acos(1-2*x(1:N/2)/c);
% Finding the midpoints of theta
thetamid = (theta(2:end)+theta(1:end-1))/2;
% Calculating the thin airfoil alpha L=0
thinalpha = 1/pi.*trapz(thetamid,dz_dx.*(cos(thetamid)-1));
% Finding the slope and y intercept of the thick airfoil data
fun = polyfit(alphas*pi/180,cl,1);
% Finding the x intercept of the thick airfoil data
alphal0 = (-fun(2)/fun(1));

% Slope of the CFD data
a = polyfit(T.AoA(6:end)*pi/180,T.Cl(6:end),1);
```

```
[maxCl,I] = max(T.Cl);
maxAlpha = T.AoA(I);
fprintf('Lift slope of the CFD data = %1.2f\n',a(1))
fprintf('Stall angle of attack = %1.2f\n',maxAlpha)
fprintf('Maximum Cl = %1.2f\n',maxCl)


thinCl = 2*pi.*alphas*pi/180;

figure
hold on
plot(alphas,cl)
plot(alphas,thinCl)
plot(T.AoA,T.Cl)
title('C_l vs. \alpha')
xlabel('\alpha')
ylabel('C_l')
legend('Vortex Panel','Thin Airfoil','RANS','location','southeast')
```

```
Lift slope of the CFD data = 5.91
Stall angle of attack = 17.00
Maximum Cl = 1.61
```



Functions Called

The following functions were built and called as part of this assignment.

```
function [x,y,yc] = NACA_Airfoils(m,p,t,c,N)
% NACA_Airfoils produces the x and y coordinates of a desired NACA
 four
% digit airfoil and the mean camber line, provided the four digits
 that
% come from m, p, and t, the chord length, and the number of points N

% Actual values of m, p, and t
m = m/100;
p = p/10;
t = t/100;
% Using the method from Kuthe and Chow to get x values
x = c/2+c/2*cos(linspace(0,pi,ceil((N+1)/2)));

% Y thickness
yt = t.*c/0.2.*(0.2969.*sqrt(x./c)-0.1260.*(x./c)-0.3516.*(x./
c).^2+...
    0.2843.*(x./c).^3-0.1036.*(x./c).^4);

% Y mean camber (This somehow works but I'm not sure if it's more
 efficient
% than a regular for loop)
for i = numel(x)
    if (0 <= x(i)) && (x(i) <= p*c)
        yc(i) = (m/p^2).*x(i).*(2*p - x(i)./c);
    else
        yc(i) = (m/(1-p)^2).*(c - x(i)).*(1 + x(i)./c - 2*p);
    end
end
% Calculating xi
yc(end) = 0;
xi =atan(diff(yc));
xi(end+1) = 0;

% Finding X upper and X lower
xU = x-yt.*sin(xi);
xL = x+yt.*sin(xi);

% Finding Y upper and Y lower then concatenating everything
yU = yc+yt.*cos(xi);
yL = yc-yt.*cos(xi);
x = [xL,fliplr(xU(1:end-1))];
y = [yL,fliplr(yU(1:end-1))];
end

function [cl,xc,yc] = Vortex_Panel(alpha,v_inf,c,x,y,M)
% Vortex_Panel uses vortex method to calculate the coefficient of lift
% around a given airfoil provided the AoA, V infinity, chord length, x
 and
% y locations of the airfoil, total number of points M, and a boolean
 value
% for whether or not to plot
```

```
MP1    = M+1;
xc     = zeros(1,M);
yc     = zeros(1,M);
S      = zeros(1,M);
theta  = zeros(1,M);
sine   = zeros(1,M);
cosine = zeros(1,M);
RHS    = zeros(1,M);
Cp     = zeros(1,M);
V      = zeros(1,M);
CN1    = zeros(M,M);
CN2    = zeros(M,M);
CT1    = zeros(M,M);
CT2    = zeros(M,M);
AN     = zeros(M,M);
AT     = zeros(M,M);

for i = 1:M
    ip1       = i+1;
    xc(i)     = 0.5*(x(i)+x(ip1));
    yc(i)     = 0.5*(y(i)+y(ip1));
    S(i)      = sqrt((x(ip1)-x(i))^2 + (y(ip1)-y(i))^2);
    theta(i)  = atan2((y(ip1)-y(i)),(x(ip1)-x(i)));
    sine(i)   = sin(theta(i));
    cosine(i) = cos(theta(i));
    RHS(i)    = sin(theta(i)-alpha);
end

for i = 1:M
    for j = 1:M
        if i == j
            CN1(i,j) = -1;
            CN2(i,j) = 1;
            CT1(i,j) = 0.5*pi;
            CT2(i,j) = 0.5*pi;

        else
            A = -(xc(i)-x(j))*cosine(j)-(yc(i)-y(j))*sine(j);
            B = (xc(i)-x(j))^2+(yc(i)-y(j))^2;
            C = sin(theta(i)-theta(j));
            D = cos(theta(i)-theta(j));
            E = (xc(i)-x(j))*sine(j)-(yc(i)-y(j))*cosine(j);
            F = log(1+S(j)*(S(j)+2*A)/B);
            G = atan2(E*S(j),B+A*S(j));
            P = (xc(i)-x(j))*sin(theta(i)-2*theta(j))+...
                (yc(i)-y(j))*cos(theta(i)-2*theta(j));
            Q = (xc(i)-x(j))*cos(theta(i)-2*theta(j))-...
                (yc(i)-y(j))*sin(theta(i)-2*theta(j));
            CN2(i,j) = D+.5*Q*F/S(j)-(A*C+D*E)*G/S(j);
            CN1(i,j) = .5*D*F+C*G-CN2(i,j);
            CT2(i,j) = C+.5*P*F/S(j)+(A*D-C*E)*G/S(j);
            CT1(i,j) = .5*C*F-D*G-CT2(i,j);
        end
    end
```

```
end

for i=1:M
    AN(i,1) = CN1(i,1);
    AN(i,MP1) = CN2(i,M);
    AT(i,1) = CT1(i,1);
    AT(i,MP1) = CT2(i,M);
    for j=2:M
        AN(i,j) = CN1(i,j)+CN2(i,j-1);
        AT(i,j) = CT1(i,j)+CT2(i,j-1);
    end
end
AN(MP1,1)   = 1;
AN(MP1,MP1) = 1;
for j = 2:M
    AN(MP1,j) = 0;
end
RHS(MP1) = 0;
gamma = AN\RHS';
for i=1:M
    V(i) = cos(theta(i)-alpha);
    for j=1:MP1
        V(i) = V(i)+AT(i,j)*gamma(j);
    end
    Cp(i) = 1-V(i)^2;
end

% Calclating Gamma
rho = 1.225;
pinf = 101325;
Qinf = .5 * rho * v_inf^2;
S = [S,S(1)];
Gamma = sum(2*pi*v_inf*(gamma'.*S));
cl = 2*Gamma/(v_inf*c);

end
```

Published with MATLAB® R2019b