Investigation & Analysis of MATLAB Code for Second Order Vortex Panel Method on NACA 2412 Airfoil

A. Aggarwal [1]
Embry-Riddle Aeronautical University, Daytona Beach, Florida, 32114

G. Greiner [2]
Embry-Riddle Aeronautical University, Daytona Beach, Florida, 32114

Abstract – The second order vortex panel method is a numerical technique used to compute the aerodynamic forces acting on a body in a fluid flow. In this study, we apply the vortex panel method to a NACA 2412 airfoil to determine the force coefficients. The NACA 2412 airfoil is a widely used airfoil shape with good lift-to-drag characteristics, making it a popular choice for a range of applications. Using the second order vortex panel method, we discretize the airfoil surface into a series of panels and model the flow around the airfoil as a series of vortex panels. By solving for the vortex strengths and satisfying the no-penetration condition on each panel, we can compute the lift, drag, and moment coefficients for the airfoil. We present the results of our computations for the NACA 2412 airfoil at various angles of attack and compare them to experimental data and results obtained using other numerical techniques. Our results demonstrate that the second order vortex panel method is capable of accurately predicting the aerodynamic forces on the airfoil, with good agreement with experimental data and other numerical methods. Overall, this study highlights the effectiveness of the second order vortex panel method as a tool for analyzing and designing airfoils, and its potential for use in a range of practical applications.

Nomenclature

yc = **camber y-ordinate**
yt = **thickness y-ordinate**
x = **x-coordinate**
m = **max camber**
p = **position of maximum camber**
Cpu = **coefficient of pressure upper surface**
Cpu = **coefficient of pressure upper surface**
Cpl = **coefficient of pressure lower surface**
Cl = **coefficient of lift**
Cd = **coefficient of drag**
Cmle = **coefficient of moment about leading edge**
Cmc/4 = **coefficient of moment about quarter chord**
Cmac = **coefficient of moment about aerodynamic center**
x_ac = **aerodynamic center**
x_cp = **center of pressure**
α = **angle of attack**
a0 **= slope of Cl and AoA**
m0 **= slope of Cmc/4 and AoA**

[1]BSc. Aerospace Engineering Candidate, Dept. of Aerospace Engineering
[2]Associate Professor, Dept. of Aerospace Engineering

I.Introduction

Vortex panel method has a wide range of application in engineering and science, including aerodynamics, hydrodynamics and oceanography. The computational tool is used to model ideal fluid-flows in which the effects of compressibility and viscosity are negligible- around solid objects The second order vortex panel technique is an advanced version of the of the vortex panel method introduced in the 1970s [1]. The approach models the flow past an airfoil as the summation of a uniform flow (same speed and direction everywhere) and a series of vortex 'panel' arranged to form a closed polygon with a shape that approximates the actual curved shape of the airfoil, see figure 1. This results in a more accurate representation of the flow felid and a more realistic simulation of the fluid dynamics.

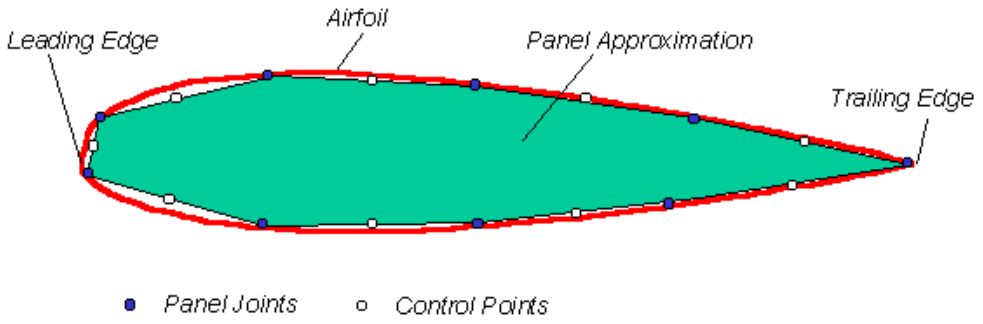

Figure 1: Vortex panel approximation for an airfoil ("The Vortex-Panel Method," Layton, W. J)

The general airfoil section is modeled by discretizing the surface contours using singularity panels. Each vortex is assigned a constant vortex strength. The vortex strength is calculated such that resulting flow satisfies the no-slip boundary condition at each panel. The velocity potential and steam function of the flow are then calculated using the superposition of the potential flow singularities on the surface. The resulting coefficients of pressure is determined. The final equations is obtained using the Kutta condition to be satisfied the strengths of the vortex panel must be equal and opposite where they meet at the trailing-edge joint [1]. The vorticity of each panel is chosen to match the tangential velocity on the panel. The resulting system of equations can be solved using linear algebra techniques to obtain the vorticity distribution and the resulting flow field. One the vorticity distribution has been calculated, the pressure distribution on the surface of the body can be obtained by integrating the normal velocity over each panel. The lift and drag forces on the body can be calculated by integrating the pressure distribution of the body [2].

The vortex panel method has many advantages over numerical methos of solving potential flow problems. Its is simle, computationally efficient and can be used to model complex 2D and 3D geometries. However, the flow neglects potential, viscous effects and assumes the flow is irrotational. This means the model flows with significant amounts of turbulence or vorticity are not effectively computed. Furthermore, the computational techniques assumes that body is immersed in an infinite fluid which mean it cannot be used model flows near boundaries or in confined spaces [3]. The following study analysis a NACA 2412 by evolving source panel method code to vortex panel and analyzing the coefficients.

II. Code & Equations

The design process for the project involved replacing the circle in the Source_Panel_Method_Circle.m- attached in Appendix- code script file to generate a NACA 4-digint airfoil ordinances. Next the boundary condition x_c(1) = 1, y_c(1) = 0 and x_c(n+1)=1, y_c(n+1) =0 where n = numbers of panels was applied. The airfoil ordinates for a NACA 2412 were used and verified by the Abbott and Von Doenhoff ordinances [4]. Next the integration variables from Kuthe and Chow, 'Foundation of Aerodynamics' [5] were used to modify the existing code. The trapezoid rule was used to integrate the Cp. Although, Simpson's rule is more accurate than the trapezoidal rule for integration of smooth and simple shapes, the trapezoidal rule is computationally simpler to implement for sharp features. The cp vs x showed sharper edges and trapz function is optimized and exists in MALAB. Next the forces and moment about quarter chord coefficients were determined. Finally, an outer loop for angle of attack was used to plot the coefficient of lift, coefficient of quarter chord, and center of pressure against angle of attack. The slopes were used to determined the aerodynamic center and the coefficient of moment about the aerodynamic center.

A. Airfoil

In order to code the ordinates for the NACA 2412 airfoil Abbott and Von Doenhoff NACA four-digit wing section was used. The x coordinates were split into two where the x1 is represent the leading edge and x2 are the ordinates form the leading circle to the trailing edge. The means lines were determined using the equations:

$$yc = \left(\frac{m}{p^2}\right)(2 * p * x1 - x1^2) \qquad forward\ of\ maximum\ ordinates$$

$$yc = \left(\frac{m}{(1-p)^2}\right)\left((1 - 2 * p) + (2 * p * x2) - x2^2\right) \quad aft\ of\ maximum\ ordinates$$

The dy for each camber was determined and then multiplied by the tan to determine the slop of the camber line. The thickness distribution for the NACA 4 digit airfoil was determined using the equation:

$$y_t = 5 * t * \left((0.29690 * sqrt(x)) - (0.12600 * x) - (0.35160 * (x^2)) + (0.28430 * (x^3)) - (0.10150.* (x^4))\right)$$

The coordinates for x and y ordinates for the upper surface and lower surface were determines using their respective trigonometric relationships. Th airfoil was oriented from trailing edge to lower surface and leading edge to upper surface and the trailing edge. The airfoil code is depicted in figure 2.

```
m = 0.02; % max camber
p=0.4; % psotion of max camber
t = 0.12;
k1 = 15.957;%???

r = 1.1019.*(t^2); %radius of leading edge circle
x1 = linspace(r/3,p,num); %x coordinates nose cicle to m
x2 = linspace(p,1,num); %x coordinates m to 1
x = ([x1 x2(:,[2:num])]);

y_cam_1 = (m/p^2).*(2*p.*x1-(x1.^2)); %camber line forward of max ordinate
y_cam_2 = (m/((1-p)^2)).*((1-2*p)+(2*p.*x2)-x2.^2); %camber line afte of maximum ordinate
y_cam = [y_cam_1 y_cam_2(:,[2:num])];

dy_cam_1 = (m/p^2).*(2*p-(2.*x1)); %derivative forward
dy_cam_2 = (m/((1-p)^2)).*(2*p-(2.*x2)); %derivative of aft
dy_cam = [dy_cam_1 dy_cam_2(:,[2:num])]; %merged derivative of camber line
theta = atan(dy_cam); %slope of camber line

y_t = 5.*t.*((0.29690.*sqrt(x))-(0.12600.*x)-(0.35160.*(x.^2)) +(0.28430.*(x.^3))-(0.10150.*(x.^4))); %thickness equation

x_upper = x-(y_t.*sin(theta)); %x coordinates of upper surface
x_lower = x+(y_t.*sin(theta)); %x coordinates of lower surface
y_upper = y_cam+(y_t.*cos(theta)); %y coordinates of upper surface
y_lower = y_cam-(y_t.*cos(theta)); %y coordinates of lower surface

x_lower = flip(x_lower);
y_lower = flip (y_lower);

```

Figure 2: Section of code to model the airfoil ordinates

B. The Cn(i,j) Matrix

The study focused on modifying the existing integration variables to optimize for the vortex panel. Equations and for loops were added to satisfy the Kuethe and Chows's stud. Han Liul converse paper was used to cross reference. The section of the code is presented in figure 3.

```
%% build the Cn(i,j) matrix
Cn2 = eye(n,n)*1;  Cn1 = eye(n,n)*-1;Ct1 = eye(n,n)*pi/2;
Ct2 = eye(n,n)*pi/2;
for i = 1:n
    for j = 1:n
        if i ~= j

            A = -(xmp(i) - X(j))*cosd(phi(j)) ...
                -(ymp(i) - Y(j))*sind(phi(j));
            C = sind(phi(i)-phi(j));
            B = (xmp(i) - X(j))^2 + (ymp(i) - Y(j))^2;
            D = cosd(phi(i) - phi(j));
            E = (xmp(i)-X(j))*sind(phi(j))-(ymp(i)-Y(j))*cosd(phi(j));
            F = log(1 + ((Sj(j))^2 + (2*A*Sj(j))) / B);
            G = atan2((E*Sj(j)) , (B + A*Sj(j)));
            P = (xmp(i)-X(j))*sind(phi(i)-2*phi(j)) + (ymp(i)-Y(j))*cosd(phi(i)-2*phi(j));
            Q = ((xmp(i) - X(j)) * cosd(phi(i) - 2*phi(j))) - ((ymp(i) - Y(j)) * sind(phi(i) - 2*phi(j)));
            % normal component
            Cn2(i,j) = D + ((0.5*Q*F)/Sj(j))-((A*C + D*E)*(G/Sj(j)));
            Cn1(i,j) = 0.5*D*F + C*G - Cn2(i,j);
            Ct2(i,j) = C + ((0.5*P*F)/Sj(j))+((A*D - C*E)*(G/Sj(j)));
            Ct1(i,j) = 0.5*C*F - D*G - Ct2(i,j);

        end
    end
end

An = zeros(n+1,n+1); RHS = zeros(n+1,1);
for i= 1:n
    An(i,1) = Cn1(i,1);
    An(i,n+1) = Cn2(i,n);
    RHS(i) = sind(phi(i)-alpha);
    for j = 2:n
        An(i,j) = Cn1(i,j) + Cn2(i,j-1);
    end
end
An(n+1,1) = 1;
An(n+1,n+1) = 1;

for j = 2:n
        An(n+1,j) = 0;
end
```

Figure 3: Shows the integration variables optimized for the vortex panel section

C. The Coefficients of Pressure

The section of code you provided involves calculating the velocity and pressure coefficients on the surface of an airfoil using the vortex panel method. The first line of the code, gamma_prime = An\RHS, computes the strengths of the vortex panels based on the system of equations derived from the panel method. An represents the matrix of coefficients in the system of equations, and RHS represents the right-hand side of the equations. The next two lines of the code initialize the velocity and pressure coefficient arrays V and cp with zeros. These arrays will be filled in with values for each panel in the subsequent loop. The loop runs over each panel i on the airfoil surface and calculates the velocity and pressure coefficients for that panel. For each panel i, the code calculates the freestream velocity component normal to the panel, V(i), as well as the contribution to the velocity due to each vortex panel j, At(i,j)*gamma_prime(j). At

represents the influence coefficient matrix, which relates the strength of each vortex panel to the velocity induced on each panel. The code then calculates the pressure coefficient for panel i based on the velocity coefficient using the formula cp(i) = 1 - (V(i))^2. The pressure coefficient is a measure of the pressure distribution on the surface of the airfoil, which is an important parameter in determining lift and drag forces.

```
34      V = zeros(n+1,1);
35      cp = zeros(n,1);
36
37      for i = 1:n
38          V(i) = cosd(phi(i)-alpha);
39          for j = 1:n+1
40          V(i) = V(i) + At(i,j)*gamma_prime(j);
41          cp(i) = 1 - (V(i))^2;
42          end
43
```

Figure 4: Shows the MATLAB script for cp

D. The Coefficients

The trapz function and the fit function of MATLAB was used to determine the all the other coefficients, slopes and locations of interests. The shear force was assumed to be negligible since viscous effects were not considered. The MATLAB script in Appendix show the general equations in the script to calculate all the value :

$$Cl = \int_0^1 Cpl - Cpu\, dx$$

$$Cmle = \int_0^1 (Cpu - Cpl)x\, dx$$

The calculation Cd required the determination of dyu/dx and the dyl/dx and to determine that a for loop was run and point i+1 was subtracted from i from the data of y and x upper and lower. The Cd theoretical formula is:

$$Cd = \int_0^1 (Cpu\, \frac{dyu}{dx} - Cpl \frac{dyl}{dx})\, dx$$

The following functions were determined through arithmetic relationships. It is to be noted that alpha, Cl and Cmc/4 values were transposed after this step to enable the use of the fit function.

$$xcp = -\frac{Cmle}{Cl}$$

$$Cmc4 = Cmle + \frac{Cl}{4}$$

$$xac = -\frac{m0}{a0} + 0.25$$

In order to determine the Cmac value any corresponding and arbitrary Cl and Cmc/4 values were used. This is because the slope of Cmac and angle of attack is zero; therefore, there is should theoretically be no difference between in calculated Cmac at any angle of attack. The Cmac was determined using the equation:

$$Cmac = Cl(xac - 0.25) + Cmc4$$

The values of the slope were determined by using the fit function and linearly fitting the values. The function syntax is fit(x, y, 'Poly1'). The slope function are:

$$m0 = \frac{\partial Cmc4}{\partial \alpha}$$

$$a0 = \frac{\partial Cl}{\partial \alpha}$$

It is to be noted that in order to use the above function xmp and ymp values were divided into upper and lower surface values and for the cd the last value was deleted for evaluation in order to determine dx and dy.

III.Results

The number of panels were increased, and the airfoil became smoother and the Cp values at the trailing edge showed better convergence. Figure 5 shows side by side airfoils plots at angle of attack equals to zero, velocity equal to one meter per second.

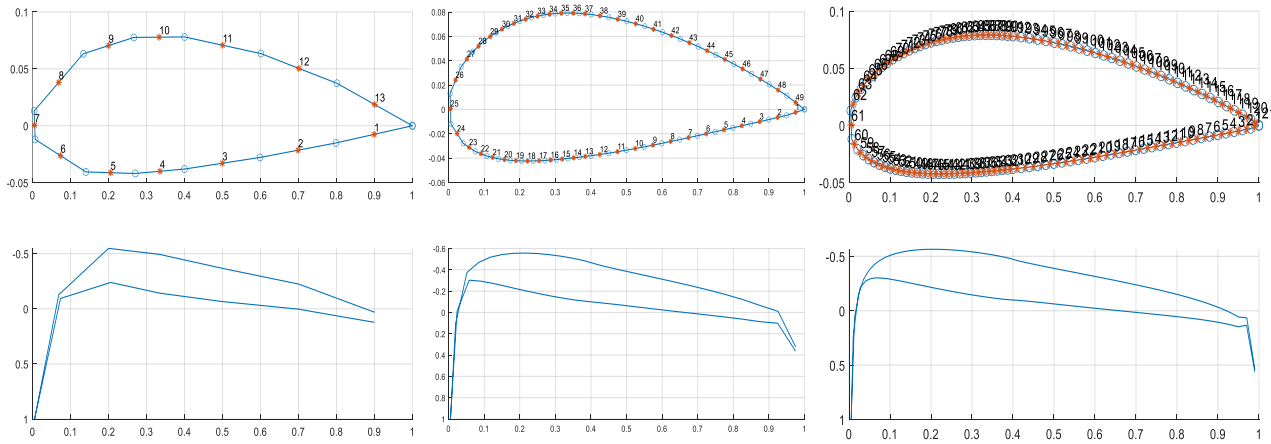

Figure 5: NACA 2412 vs x and Cp vs x for number of panel 12,48 and 120 respectively (left to right)

The determined coefficients and location of center of pressure for inviscid flow at angle of attack of zero and velocity of one meter per second for panel 12, 48 and 120 are summarized in Table 1.

	Panel No. 12	Panel No. 48	Panel No. 120
Cl	0.2057	0.2486	0.2552
Cd	0.0086	-0.0051	-0.0118
Cmle	-0.0936	-0.1145	-0.1182
Cmc/4	-0.0421	-0.0524	-0.0544
xcp	0.4548	0.4608	0.4632

Table 1: The determined coefficients for location of center of pressure

Comparing the results with Abbot and Von Doeff, it is to be noted with increased number of panne the Cl value gets closer to the experimentally determined values-0.25; however, all other valued vary with approximately 30% error from Abbot and Van Doeff values for NACA 2412.

Next an outer loop for angle of attack was added to the code and the graphs for cl, Cmc4 and xcp were plotted. Forty eight panels were selected since the Cl value was the closest to the experimentally determined value.

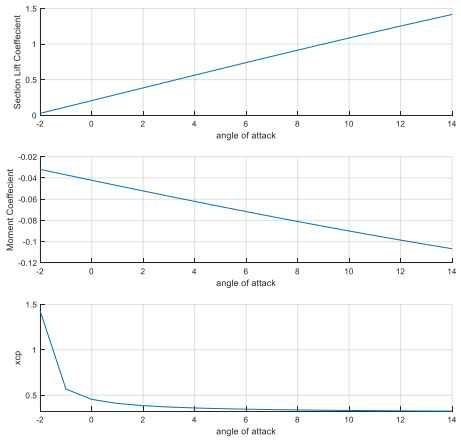

Figure 6: Shows Cl, Cmc4 and xcp vs AoA for NACA 2412 with 48 number of panel

Table two shows the values for the vortex panel method as well as the Perkin and Hage [6] values and the percent of error.

	Vortex Panel Method Code	Perkin and Hage Airfoil Tables	Percent Error
a0	0.1057	0.104	1.6%
m0	-0.0029	-	-
xac	0.2771	0.247	12%
cmac	-0.0457	-0.047	2.8%

Table 2: Shows the values obtained for NACA 2412 48 panels and Perkin and Hage

IV.Conclusion

The results reflect that the Vortex Panel Method if effective in determining the Cl and relatively close in determine a0, m0 and aerodynamic center and moment about aerodynamic center. However, the data reflects that second order vortex panel method does not accurately predict drag coefficients. This is because the method assumes the flow is inviscid and neglects the effects of viscous forces and boundary layer separation. The method also assumes that the vortex filaments are infinitely thin and that the velocity induced by the vortices is constant across the width of the panel. In reality, the vortices have a finite thickness, and the induced velocity can vary across the width of the panel which can lead to inaccuracies in the computed forces. Thus, second order vortex panel method is useful in providing preliminary estimates of aerodynamic forces and pressure distribution but should be used with caution and the assumptions should be taken into account.

Based on the analysis provided future software engineers coding the second-order vortex panel method are encouraged to use higher-order interpolation, incorporate boundary layer effects and implement correction factor for drag . The vortex panel method underestimates Cd and using correction factor such as Schrenk's approximation can improve the overall accuracy of the method. Moreover, incorporating boundary layer effects such as turbulence model can assimilate viscous effects and can increase Cd values. Finally, use Gauss-Lengdre quadrature to compute the influence of each vortex panel on each other is favorable over numerical integration. A more accurate numerical integration scheme can be implemented in advanced languages and libraries such as python's SciPy and NumPy as well as Jullia's QuadGk and Cubature.

Appendix

Vortex Panel Code

```
clc; clear

alpha_value = linspace(-2,14,(17));
cl_value = zeros(1,length(alpha_value));
cm_c_four_value = zeros(1,length(alpha_value));
xcp_value = zeros(1,length(alpha_value));

for i_final = 1:length(alpha_value)
alpha = alpha_value(i_final);

V = 1;
num = 8;

%% Body input section, unit radius
% panel = [1, 2, 3, 4 ,5 ...

m = 0.02; % max camber
p=0.4; % psotion of max camber
t = 0.12;
k1 = 15.957;%???

r = 1.1019.*(t^2); %radius of leading edge circle
x1 = linspace(r/3,p,num); %x coordinates nose cicle to m
x2 = linspace(p,1,num); %x coordinates m to 1
x = ([x1 x2(:,[2:num])]);

y_cam_1 = (m/p^2).*(2*p.*x1-(x1.^2)); %camber line forward of max ordinate
y_cam_2 = (m/((1-p)^2)).*((1-2*p)+(2*p.*x2)-x2.^2); %camber line afte of maximum
ordinate
y_cam = [y_cam_1 y_cam_2(:,[2:num])];

dy_cam_1 = (m/p^2).*(2*p-(2.*x1)); %derivative forward
dy_cam_2 = (m/((1-p)^2)).*(2*p-(2.*x2)); %derivative of aft
dy_cam = [dy_cam_1 dy_cam_2(:,[2:num])]; %merged derivative of camber line
theta = atan(dy_cam); %slope of camber line

y_t = 5.*t.*((0.29690.*sqrt(x))-(0.12600.*x)-(0.35160.*(x.^2)) +(0.28430.*(x.^3))-
(0.10150.*(x.^4))); %thickness equation

x_upper = x-(y_t.*sin(theta)); %x coordinates of upper surface
x_lower = x+(y_t.*sin(theta)); %x coordinates of lower surface
y_upper = y_cam+(y_t.*cos(theta)); %y coordinates of upper surface
y_lower = y_cam-(y_t.*cos(theta)); %y coordinates of lower surface

x_lower = flip(x_lower);
y_lower = flip (y_lower);

X = [x_lower x_upper];
Y = [y_lower y_upper];
X(1) = 1;
```

```
Y(1) = 0;
X(end) = 1;
Y(end) = 0;


% vortex panel length, mid-point and orientation angle
n = length(X)- 1;

Sj = zeros(n,1); phi = zeros(n,1);
xmp = zeros(n,1); ymp = zeros(n,1);
for i = 1:n
    % Length of each panel
    Sj(i) = sqrt((X(i+1) - X(i))^2 + (Y(i+1) - Y(i))^2);
    % mid-point of each panel
    xmp(i) = 0.5*(X(i+1) + X(i));
    ymp(i) = 0.5*(Y(i+1) + Y(i));
    phi(i) = atan2d( (Y(i+1) - Y(i)),((X(i+1) - X(i))) );
end

%
% subplot(2,1,1);hold on;
% plot(X,Y,'o-'); plot(xmp,ymp,'*'); label_1 = string(1:n);
% text(xmp,ymp,label_1,'VerticalAlignment','bottom');
% grid on; hold off;


%% build the Cn(i,j) matrix
Cn2 = eye(n,n)*1;  Cn1 = eye(n,n)*-1;Ct1 = eye(n,n)*pi/2;
Ct2 = eye(n,n)*pi/2;
for i = 1:n
    for j = 1:n
        if i ~= j

            A = -(xmp(i) - X(j))*cosd(phi(j)) ...
                -(ymp(i) - Y(j))*sind(phi(j));
            C = sind(phi(i)-phi(j));
            B = (xmp(i) - X(j))^2 + (ymp(i) - Y(j))^2;
            D = cosd(phi(i) - phi(j));
            E = (xmp(i)-X(j))*sind(phi(j))-(ymp(i)-Y(j))*cosd(phi(j));
            F = log(1 + ((Sj(j))^2 + (2*A*Sj(j))) / B);
            G = atan2((E*Sj(j)) , (B + A*Sj(j)));
            P = (xmp(i)-X(j))*sind(phi(i)-2*phi(j)) + (ymp(i)-Y(j))*cosd(phi(i)-
2*phi(j));
            Q = ((xmp(i) - X(j)) * cosd(phi(i) - 2*phi(j))) - ((ymp(i) - Y(j)) *
sind(phi(i) - 2*phi(j)));
            % normal component
            Cn2(i,j) = D + ((0.5*Q*F)/Sj(j))-((A*C + D*E)*(G/Sj(j)));
            Cn1(i,j) = 0.5*D*F + C*G - Cn2(i,j);
            Ct2(i,j) = C + ((0.5*P*F)/Sj(j))+((A*D - C*E)*(G/Sj(j)));
            Ct1(i,j) = 0.5*C*F - D*G - Ct2(i,j);

        end
    end
end
```

```
An = zeros(n+1,n+1); RHS = zeros(n+1,1);
for i= 1:n
    An(i,1) = Cn1(i,1);
    An(i,n+1) = Cn2(i,n);
    RHS(i) = sind(phi(i)-alpha);
    for j = 2:n
        An(i,j) = Cn1(i,j) + Cn2(i,j-1);
    end
end
An(n+1,1) = 1;
An(n+1,n+1) = 1;

for j = 2:n
        An(n+1,j) = 0;
end
RHS(n+1) = 0;

At = zeros(n+1,n+1);
for i=1:n
    At(i,1) = Ct1(i,1);
    At(i,n+1) = Ct2(i,n);
    for j =2:n
        At(i,j) = Ct1(i,j) + Ct2(i,j-1);
    end


end
gamma_prime = An\RHS;

V = zeros(n+1,1);
cp = zeros(n,1);

for i = 1:n
    V(i) = cosd(phi(i)-alpha);
    for j = 1:n+1
    V(i) = V(i) + At(i,j)*gamma_prime(j);
    cp(i) = 1 - (V(i))^2;
    end

end
% subplot(2,1,2); hold on;
% plot(xmp,cp);
% set(gca,'Ydir','reverse');
% %text(theta_p1,Cp_p1,label_1,'VerticalAlignment','bottom');
% grid on; hold off


xmp_lower = xmp(1:(length(xmp))/2);
cp_lower = flip(cp(1:(length(cp))/2));
ymp_lower = flip (ymp(1:(length(xmp))/2));

xmp_upper = xmp(((length(xmp))/2)+1:end);
ymp_upper = ymp(((length(xmp))/2)+1:end);

cp_upper = cp(((length(xmp))/2)+1:end);
```

```
dcp = cp_lower-cp_upper;

dy_upper = zeros(length(ymp_upper)-1,1);
dy_lower = zeros(length(ymp_upper)-1,1);
dx_upper = zeros(length(ymp_upper)-1,1);
dx_lower = zeros(length(ymp_upper)-1,1);



for i=1:(length(ymp_upper)-1)
    dy_upper(i) = ymp_upper(i+1)-ymp_upper(i);
    dy_lower(i) = ymp_lower(i+1)-ymp_lower(i);
    dx_upper(i) = xmp_upper(i+1)-xmp_upper(i);
    dx_lower(i) = xmp_lower(i+1)-xmp_lower(i);
end
dyu_dx = dy_upper./dx_upper;
dyl_dx = dy_lower./dx_upper;

cd_cp_upper = cp_upper;
cd_cp_lower = cp_lower;
cd_xmp_upper = xmp_upper;

cd_cp_upper(end)=[];
cd_cp_lower(end) = [];
cd_xmp_upper (end) =[];


cd_integral_value = (cd_cp_upper.*dyu_dx) - (cd_cp_lower.*dyl_dx);
cd = trapz(cd_xmp_upper,cd_integral_value);
cl = trapz(xmp_upper,dcp);
cmle = trapz(xmp_upper,-dcp.*xmp_upper);
xcp = -cmle/cl;
cm_c_four = cmle +(cl/4);

cl_value(i_final) = cl;
cm_c_four_value(i_final) = cm_c_four;
xcp_value(i_final) = xcp;
end
alpha_value = transpose(alpha_value);
cl_value = transpose(cl_value);
cm_c_four_value= transpose(cm_c_four_value);



subplot(3,1,1); hold on;
plot(alpha_value,cl_value);
xlabel("angle of attack")
ylabel("Section Lift Coeffecient")

grid on; hold off
```

```
subplot(3,1,2); hold on;
plot(alpha_value,cm_c_four_value);
xlabel("angle of attack")
ylabel("Moment Coeffecient")
%text(theta_p1,Cp_p1,label_1,'VerticalAlignment','bottom');grid on; hold off
grid on; hold off

subplot(3,1,3); hold on;
plot(alpha_value,xcp_value);
xlabel("angle of attack")
ylabel("xcp")
%text(theta_p1,Cp_p1,label_1,'VerticalAlignment','bottom');
grid on; hold off

a_zero_fit = fit(alpha_value,cl_value,'poly1');
p=coeffvalues(a_zero_fit);
a_zero = p(1);
cm_slope = fit(alpha_value,cm_c_four_value,'poly1');
p=coeffvalues(cm_slope);
m_zero = p(1);

x_ac = (-m_zero/a_zero)+0.25;
cm_ac = cl_value(3)*(x_ac-0.25
```

References

[1] Layton, W. J. (2001). The Vortex-Panel Method. Virginia Tech. Retrieved May 2, 2023, from https://www.engapplets.vt.edu/fluids/vpm/vpminfo.html

[2] "Aerodynamics for Students - Subsonic Aerofoil and Wing Theory: 2D Panel Methods." Aerodynamics for Students, 2019, www.aerodynamics4students.com/subsonic-aerofoil-and-wing-theory/2d-panel-methods.php.

[3] Katz, J., and Plotkin, A. (2001). Low-Speed Aerodynamics. 2nd ed. Cambridge, UK: Cambridge University Press..

[4] Abbott, I. H. and von Doenhoff, A. E. (1959). Theory of Wing Sections. Dover Publications.

[5] Kuethe, A. M. and Chow, C.-Y. (1998). Foundations of Aerodynamics. 5th ed. New York: John Wiley & Sons.

[6] C. D. Perkins and R. E. Hage, "Airplane Performance, Stability and Control", John Wiley & Sons, 1949.