Joel Lubinitsky AEE 342
02/20/15 Project 2a

AEE 342: Aerodynamics, Project 1c – Analysis of Symmetric Airfoils

Submitted: 02/20/15

In the study of computational fluid dynamics, there are numerous ways to model the interaction between a physical surface and the flow being modelled. In relatively simple cases, point singularities can be modelled with certain strengths in order to either ‘attract’ or ‘repel’ the flow. The strengths of these points are constrained by the system’s required geometry and flow conditions. In the current investigation, a similar method is adopted but with sources defined over finite lengths on the surface of an airfoil. These sources are known as ‘panels’ and the general name for the method is the source panel method. Here, the position and orientation of the panels are constrained by the airfoil’s geometry and the strengths are constrained by both the geometry and the flow conditions. In this part of the investigation, particular attention is paid to the process of defining the boundaries of a NACA 4-series airfoil and fitting panels to fit those boundaries. Additionally, some analysis of the airfoil’s performance is evaluated as well.

The shapes of NACA 4-series airfoils is given by a four number designation of the form $mptt$. Here, m is the maximum camber as a percent of chord length, p is the position along the chord of maximum camber as tenths of chord length, and tt is the thickness as a percent of chord length. Two general categories of airfoil include those that are symmetric, and those that are asymmetric. This can be determined by the first two numbers in the airfoil designation. If m and p are both 0, then the airfoil is symmetric. Otherwise, it has some camber. For either case, a set of x-values must first be defined from which to plot the thickness and curvature of the particular airfoil. These can be defined well by the relation

$$x = \frac{1}{2}(1 + \cos\zeta)$$

Where ζ can be a set of values chosen from π to 2π, so that resulting values of x are between 0 and 1. This trigonometric definition of x-values generally leads to better results than those of a linear distribution because it causes points to be plotted at a greater density near the ends of the airfoil than in the middle. This is desirable because many of the important flow interactions occur on the leading and trailing edges of the airfoil and would thus demand greater precision. A lower order of precision is generally acceptable near the middle of the airfoil. Next, the camber line must be defined. This is given for a cambered NACA 4-series airfoil as the piecewise function

$$y_c = \begin{cases} \dfrac{m}{p^2}(2px - x^2) & x \le p \\ \dfrac{m}{(1-p)^2}[(1-2p) + 2px - x^2] & x \ge p \end{cases}$$

In the case of a symmetric airfoil, the camber line coincides with the chord exactly, and is thus simply a straight line from 0 to 1 along the x-axis. Next, the NACA 4-series thickness is given by

$$y_t = \frac{tt}{0.20}\left(0.2969\sqrt{x} - 0.1260x - 0.3516x^2 + 0.2843x^3 - 0.1015x^4\right)$$

The thickness of the airfoil is taken at an angle perpendicular to the camber, so a trigonometric transformation must be performed to find the actual locations of points on the airfoil. However, in the case of an uncambered airfoil, this is trivial since the direction of thickness lines up exactly with the Cartesian y-axis. Therefore, the locations of points on a symmetric airfoil are given by

$$x_{U,\ symmetric} = x$$

$$x_{L,\ symmetric} = x$$

$$y_{U,\ symmetric} = y_t$$

$$y_{L,\ symmetric} = -y_t$$

The angle of the camber line can be found by taking the arctan of its slope, as seen in the relation

$$\theta = \tan^{-1}\frac{dy_c}{dx}$$

With this angle known at all points, the points on the surface of a cambered airfoil can be determined by

$$x_U = x - y_t \sin\theta$$

$$x_L = x + y_t \sin\theta$$

$$y_U = y_c + y_t \cos\theta$$

$$y_L = y_c - y_t \cos\theta$$

With these points all defined, the geometry of the problem has been fully constrained. The next step is to generate panels connecting each of the points on the airfoil’s surface to the next one in a clockwise direction. Then, by evaluating a matrix of strengths relative to each other such that the net source of the system is equal to 0, the strengths of each source panel can be determined. Using these methods, the following solutions and visualizations have been generated. Among the graphs plotted, important values such as pressure coefficients have been determined for the airfoil. These values are particularly useful because coefficients of lift and drag can ultimately be calculated by integrating these values along the surface of the airfoil, given by

$$C_n = \int_0^1 \left(C_{p_l} - C_{p_u}\right) dx$$

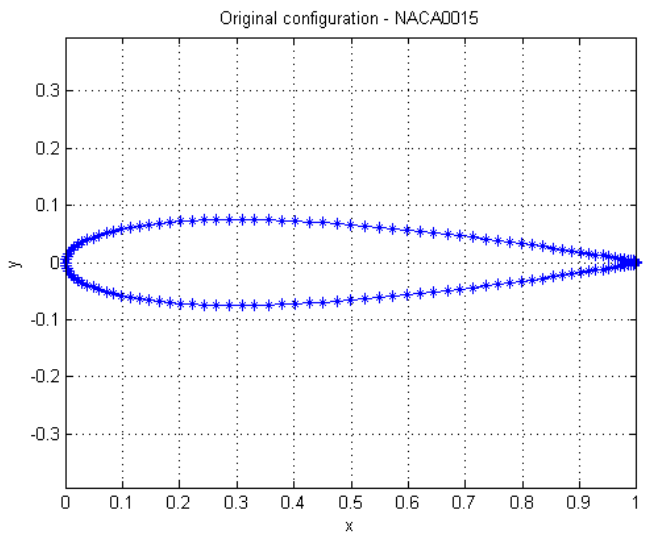
$$C_a = \int_0^1 \left(C_{p_u}\frac{dy_u}{dx} - C_{p_l}\frac{dy_l}{dx}\right) dx$$

For inviscid flow over an airfoil of chord length $c = 1$. These values can be converted to components of lift and drag by the relations

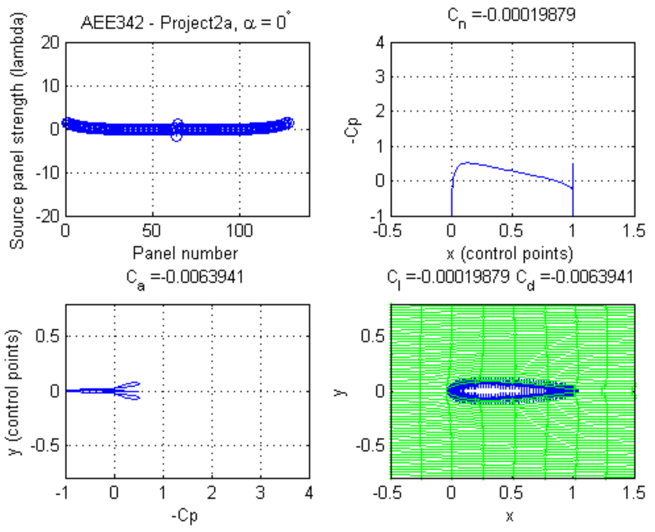
$$C_l = C_n \cos\alpha - C_a \sin\alpha$$

$$C_d = C_n \sin\alpha + C_a \cos\alpha$$

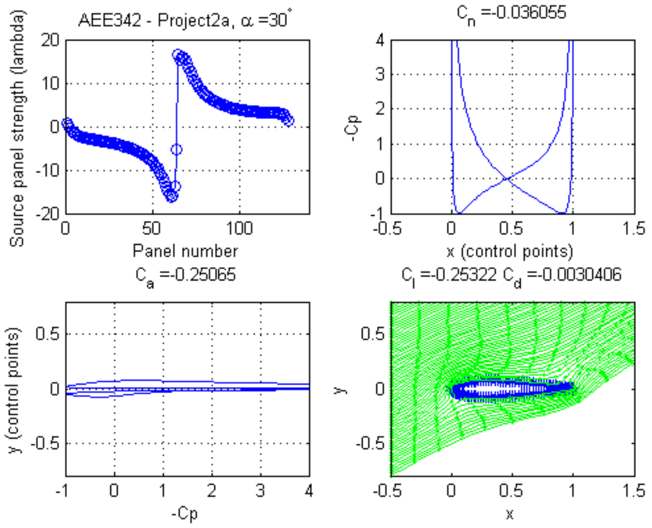
In the figure shown below, the plotted geometry of a NACA 0015 airfoil can be seen. This airfoil is clearly symmetric, and must be so since its designation states that there is 0 camber. The designation also states that the thickness of the airfoil is $0.15c$ or just 0.15. Simply by inspection, the geometry appears to match the technical specification of the airfoil well, since the maximum thickness appears to go out to about 0.7 in either y-direction. Also, the point distribution appears to have come out as desired, with the greatest concentration of points at the leading and trailing edges. This helps make the best use of a relatively low number of panels, since the airfoil surface looks nearly continuous on either end, but appears slightly coarse on the top and bottom surfaces near the middle. This particular model is composed of 128 panels.



The following figure is the analysis of this airfoil for an angle of attack $\alpha = 0$. The first subplot of this set, in the top-right, plots the source strength of each panel used. This is evaluated for the same model as above, so the number of panels remains 128, as it will for all subsequent figures. The source distribution here appears very linear, which is consistent with the symmetric geometry of the airfoil. The small jump in the middle of the plot should also be noted. The panels in this region are near the trailing edge of the airfoil. Since stagnation is expected here, this jump is reasonable since relatively strong

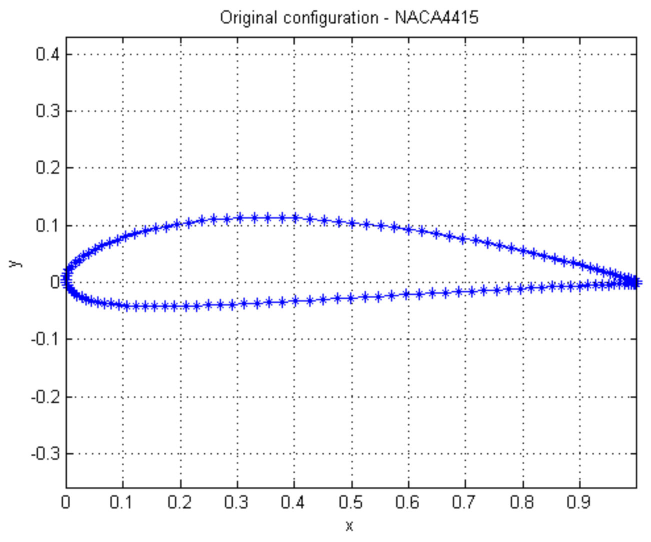


sources would be needed to influence the flow in this way. Corresponding sinks of equal magnitude would be needed locally as well in order to maintain the net source of the airfoil. The second and third subplots display the pressure coefficient as a function of x and of y, respectively. The second subplot should appear as two separate curves; one for the upper surface and one for the lower. However, in the case of a symmetric airfoil, these curves overlap since the pressure distribution is same on the top and on the bottom. This is reaffirmed by the third subplot, where the pressure distribution is clearly mirrored about $y = 0$. Finally, the fourth subplot shows the streamlines near the airfoil. This visualization is very helpful for developing a stronger intuition about the flow behavior as it interacts with the airfoil. This plot will tend to reveal more when an angle of attack is introduced. It should also be noted that calculated coefficients of normal force, axial force, lift, and drag are shown above subplots two, three, and four. These values are very small and within the range of error involved with numerical approximation. If this is in fact purely a result of error, it would suggest that the airfoil generates no lift. This would be consistent with theory because the flow involves no vorticity, which is a necessary flow parameter for the generation of lift. These values will be evaluated for other angles of attack and another airfoil to test the validity of these claims.

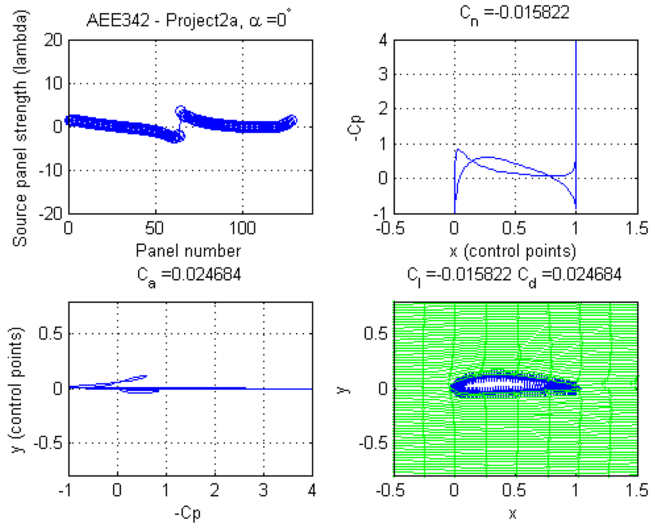


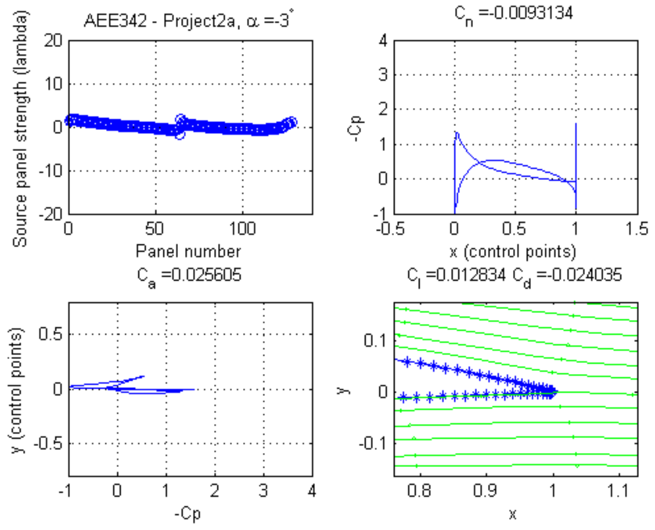
The figure shown to the right contains all the same elements as the figure discussed above, but is calculated for an angle of attack of 30 degrees. Immediately, the first subplot appears very nonlinear, with sweeping curves and a very large jump. This suggests a very large difference in source strengths between the surface facing toward the flow and the surface facing away from the flow. Specifically, the upper surface contains very strong sinks, while the lower surface contains very strong sources. The following two subplots are also very different from those of the previous figure. Here, the two pressure curves are very distinguishable. Using information from other plots of this model, it can be inferred that the curve peaking on the left side of the second subplot is upper surface, and the one on the right is the lower surface. The third plot in this figure is very difficult to interpret, but that is likely because a rather large variation in pressure coefficient was plotted against a rather low variation in y. Finally, the streamline plot shows the freestream emanating at a 30 degree angle to the airfoil chord. Some curvature can be seen on the leading and trailing edges of the airfoil due to pressure differences on the top and bottom. Also, the lines made by ‘bubbles’ help to show the differences in velocity of the flow in different parts of the freestream.

This next figure shows the plotted geometry of a NACA 4415 airfoil constructed with 128 panels. This figure matches the parameters expected from the airfoil designation. A camber of 0.04 seems like it is reasonable for this figure, although it is difficult to be certain just by inspection. The position of max camber certainly seems to be at 0.4, and the top and bottom of the airfoil appear to be at y-values of approximately 0.12 and -0.03, respectively. This clearly suggests that the thickness matches and that the overall geometry is consistent with the intended parameters.



The figure to the right displays an analysis similar to that of the NACA 0015 airfoil. However, even though the airfoil is at an angle of attack of 0, the graphs are still asymmetrical. This is because the airfoil is cambered and thus asymmetrical. The jump in the source distribution and lack of symmetry in pressure coefficients here can be interpreted the same way as those of the symmetric airfoil at an angle of attack greater than 0. That is, there are likely pressure differences between the lower and upper surfaces of the airfoil. This would also suggest that a cambered airfoil at an angle of attack of zero may behave similarly to an uncambered airfoil at some angle of attack greater than 0.





Finally, the bottom figure is again the same plot except for an angle of attack of -3. This specific angle of attack was determined iteratively with the intention of finding a condition in which the flow leaves the trailing edge in straight lines, without curling back on either surface. In this case, the source distribution appears to be very similar to that of the uncambered airfoil at an angle of attack of 0. This is because there are not very large pressure differences between the lower and upper surfaces at the leading or trailing edges. The streamline plot is zoomed in on the trailing edge to show the streamlines as they leave the upper and lower surfaces. It is clear that curvature is minimal in the streamlines. The significance of this condition is that it is likely the angle of zero lift. This is inferred from experience with other airfoils having angles of zero lift at similar values, and due to the fact that such a condition would result in minimal vorticity had the circulation been modelled in this system. Another conclusion that can be drawn, in comparing the various analyses, is that neither airfoil generates lift. This notion was brought up in the discussion of the original airfoil at an angle of attack of 0. Although other conditions did show some variation in lift and drag coefficients, they were mostly within the range of reasonably being due to numerical error. With further study, this could suggest that an airfoil and any angle of attack would not generate lift without the presence of vortices. This notion could be tested in the future by superimposing vortex sheets and by testing different configurations of parameters. One conclusion that cannot be made, however, is on the accuracy of the method of source panels for modelling flow conditions. Since no table values were used for comparison, the results of this investigation cannot be used for any absolute values. Values obtained in this investigation must be used relatively to other values in this investigation until the method used is proven effective with respect to physical testing.

Calculations

```
function Project2a(alpha, n_pan, m, p, tt)
    % Project2jfd - 2D potential flow via source panel method
    %    written by John Dannenhoffer for AEE342 (Spring 2015)
    %    adapted from Anderson's "Fundamentals of Aerodynamics, 5th ed"
    %    edited by Joel Lubinitsky for airfoil
    %
    % alpha  = angle of attack        (deg)
    % series = NACA series designator (mptt)
    % n_pan  = number of panels

    n_sl  =   60;                 % number of streamlines
    dtbub = 0.25;                 % time increment between bubbles
    xmin  = -0.5;                 % mininum X in domain
    xmax  = +1.5;                 % maximum X in domain
    ymin  = -3.0;                 % minimum Y in domain
    ymax  = +3.0;                 % maximum Y in domain

    % format inputs
    n_pan = n_pan + 2;
    m = m * 0.01;
    p = p * 0.1;
    tt = tt * 0.01;

    % define airfoil geometry
    zeta = linspace(pi, 2 * pi, n_pan / 2);
    xCamber = 0.5 * (1 + cos(zeta));
    yThickness = (tt / 0.20) * ((0.2969 * sqrt(xCamber)) - (0.1260 .*
xCamber) - (0.3516 .* xCamber .^ 2) + (0.2843 .* xCamber .^ 3) - (0.1015 .*
xCamber .^ 4));

    if p > 0 && m > 0
    indexP = max(find(xCamber < p));

    xCamber1 = xCamber(1 : indexP);
    xCamber2 = xCamber(indexP + 1 : end);
    yCamber1 = (m / p ^ 2) * (2 * p * xCamber1 - xCamber1 .^ 2);
    yCamber2 = (m / (1 - p) ^ 2) * (1 - (2 * p) + (2 * p * xCamber2) -
xCamber2 .^ 2);
    yCamber = [yCamber1, yCamber2];

    dydxCamber1 = (2 * m / p) * (1 - xCamber1 / p);
    dydxCamber2 = (2 * m / (1 - p) ^ 2) * (p - xCamber2);
    dydxCamber = [dydxCamber1, dydxCamber2];
    theta = atan(dydxCamber);

    xUpper = xCamber - yThickness .* sin(theta);
    xUpper = xUpper(1 : 1 : end - 1);
    xLower = xCamber + yThickness .* sin(theta);
    xLower = xLower(end : -1 : 2);

    yUpper = yCamber + yThickness .* cos(theta);
    yUpper = yUpper(1 : 1 : end - 1);
    yLower = yCamber - yThickness .* cos(theta);
    yLower = yLower(end : -1 : 2);

    X = [xUpper, xLower];
    Y = [yUpper, yLower];

    elseif p == 0 && m == 0
        xUpper = xCamber;
        xLower = xCamber;

        yUpper = yThickness;
        yLower = -yThickness;

        X = [xUpper(1 : 1 : end - 1), xLower(end : -1 : 2)];
        Y = [yUpper(1 : 1 : end - 1), yLower(end : -1 : 2)];

    end

    % plot the configuration with first point repeated (figure 1)
    figure(1)
    plot([X, X(1)], [Y, Y(1)], '-*')
    title(strcat('Original configuration - NACA ', num2str(m * 100),
num2str(p * 10), num2str(tt * 100)))
    xlabel('x')
    ylabel('y')
    axis equal
    grid on

    % define the freestream velocities in the x- and y- directions
    Uinf  = cos(alpha * pi/180);
    Vinf  = sin(alpha * pi/180);

    % find the source panel strengths
    [xcp, ycp, Cp, lambda] = sourcePanel(Uinf, Vinf, X, Y);

    % integrate pressure coefficient for force coefficients
    CpUpper = Cp(1 : round(n_pan / 2) - 1);
    CpLower = Cp(round(n_pan / 2) : end);
    CpLower = CpLower(end : -1 : 1);

    xCpUpper = xcp(1 : round(n_pan / 2) - 1);
    xCpLower = xcp(round(n_pan / 2) : end);
    xCpLower = xCpLower(end : -1 : 1);

    Cn = Project2aCn(n_pan, xCpUpper, xCpLower, CpUpper, CpLower);
    Ca = Project2aCa(n_pan, xCpUpper, xCpLower, CpUpper, CpLower, yUpper,
yLower);

    Cl = Cn * cos(alpha) - Ca * sin(alpha);
    Cd = Cn * sin(alpha) + Ca * cos(alpha);

    % plot the source panel strengths (figure 2, subplot 1)
    figure(2)
    subplot(2,2,1)
        plot(1:length(lambda), lambda, '-o')
        title(strcat('AEE342 - Project2a, \alpha = ', num2str(alpha),
'^\circ'))
        xlabel('Panel number')
        ylabel('Source panel strength (lambda)')
        axis([0 140 -20 20])
        grid on

    % plots the surface Cp distribution (figure 2, subplots 2 and 3)
    subplot(2,2,2)
        plot([xcp, xcp(1)], -[Cp, Cp(1)])

        title(strcat('C_n =', num2str(Cn)))
        xlabel('x (control points)')
        ylabel('-Cp')
        xlim([-0.5 1.5])
        ylim([-1 4])
        grid on

    subplot(2,2,3)
        plot(-[Cp, Cp(1)], [ycp, ycp(1)])

        title(strcat('C_a =', num2str(Ca)))
        xlabel('-Cp')
        ylabel('y (control points)')
        ylim([-0.8 0.8])
        xlim([-1 4])
        grid on

    % plot of configuration and streamlines (figure 2, subplot 4)
    subplot(2,2,4)
        plot([X, X(1)], [Y, Y(1)], '-*b')

        hold on
        for i_sl = 1 : n_sl
            xbeg = -0.5;
            ybeg = -0.8 + (i_sl-1) / (n_sl-1) * (0.8 - -0.8);

            [t_sl, xy_sl] = ode45(@(t, xy) strmLine(t, xy, Uinf, Vinf, ...
                                  X, Y, lambda), [0, 10], [xbeg, ybeg]);
            plot(xy_sl(:,1), xy_sl(:,2), '-g')

            % add bubbles to the streamlines (if desired)
            if (dtbub > 0)
                t_bub = 0 : dtbub : t_sl(end);
                x_bub = interp1(t_sl, xy_sl(:,1), t_bub);
                y_bub = interp1(t_sl, xy_sl(:,2), t_bub);
                plot(x_bub, y_bub, 'og', 'MarkerSize', 2)
            end % if
        end % for i_sl
        hold off

        title(strcat('C_l =', num2str(Cl), ' C_d =', num2str(Cd)))
        xlabel('x')
        ylabel('y')
        axis([-0.5 1.5 -0.8 0.8])

end % function Project2jfd

%--------------------------------------------------------------------

function [xcp, ycp, Cp, lambda] = sourcePanel(Uinf, Vinf, X, Y)
    % sourcePanel - source panel method in two dimensions
    %    written by John Dannenhoffer
    %
    % inputs:
    %    Uinf       x-component of freestream velocity
    %    Vinf       y-component of freestream velocity
    %    X          x-coordinate of boundary points
    %    Y          y-coordinate of boundary points
    % outputs:
    %    xcp        x-coordinate of control points
    %    ycp        y-coordinate of control points
    %    Cp         pressure coefficient at control points
    %    lambda     column-vector of panel strengths

    toler = 0.000001;                       % minimum panel length

    n = length(X);

    % find the control point locations (xcp, ycp), panel lengths (s),
    %    and panel inclinations (phi)
    xcp = zeros(1,n);
    ycp = zeros(1,n);
    S   = zeros(1,n);
    phi = zeros(1,n);

    for i = 1 : n
        if (i < n)
            ip1 = i + 1;
        else
            ip1 = 1;
        end % if

        xcp(i) =                            (X(ip1) + X(i))/2;
        ycp(i) =       (Y(ip1) + Y(i))/2;
        S(  i) = sqrt( (Y(ip1) - Y(i))^2 + (X(ip1) - X(i))^2);
        phi(i) = atan2((Y(ip1) - Y(i))    , (X(ip1) - X(i))  );

        % check that the points are distinct
        if (S(i) < toler)
            error('Points %d and %d are not distinct', i, ip1)
        end % if
    end % for i

    % set up the Mnorm matrix (which is called "I" in Anderson) and
    %    Mtang matrix (which is called "I'" in Anderson)
    Mnorm = zeros(n,n);
    Mtang = zeros(n,n);
    RHS   = zeros(n,1);
    % normal velocity at ith control point
    for i = 1 : n
        RHS(i) = -2 * pi * (Vinf * cos(phi(i)) - Uinf * sin(phi(i)));

        % contribution of jth panel
        for j = 1 : n
            if (i == j)
                Mnorm(i,j) = pi;
                Mtang(i,j) = 0;
            else
                A = - (xcp(i) - X(j)) * cos(phi(j)) - (ycp(i) - Y(j)) *
sin(phi(j));
                B =   (xcp(i) - X(j)) ^ 2           + (ycp(i) - Y(j)) ^ 2;
                D = - (xcp(i) - X(j)) * sin(phi(i)) + (ycp(i) - Y(j)) *
cos(phi(i));
                E =   (xcp(i) - X(j)) * sin(phi(j)) - (ycp(i) - Y(j)) *
cos(phi(j));
                C = sin(phi(i) - phi(j));

                Mnorm(i,j) =    C/2        * log((S(j)^2 + 2*A*S(j) + B) / B)
...
                           + (D-A*C)/   E  * (atan((S(j)+A)/E) - atan(A/E));
                Mtang(i,j) = (D-A*C)/(2*E) * log((S(j)^2 + 2*A*S(j) + B) / B)
...
                           -    C          * (atan((S(j)+A)/E) - atan(A/E));
            end % if
        end % for j
    end % for i

    % solve for the source strengths
    lambda = Mnorm \ RHS;

    % compute the tangential velocities and hence the Cp
    V = Uinf*cos(phi) + Vinf*sin(phi);
    for j = 1 : n
        V = V + lambda(j) ./ (2*pi) .* Mtang(:,j)';
    end % for
    Cp = 1 - V.^2;

    check = sum(S*lambda);
    if (abs(check) > 1.0e-10)
        fprintf(1, 'sum of sources = %f (should be 0)\n', check);
    end % if
end % function sourcePanel

%---------------------------------------------------------------------

function u = Uvel(x, y, Uinf, X, Y, lambda)
    % Uvel - x-component of velocity at (x,y)

    % uniform flow
    u = Uinf * ones(size(x));

    % loop through source panels
    n = length(lambda);
    for j = 1 : n
        if (j < n)
            jp1 = j + 1;
        else
            jp1 = 1;
        end % if

        S   = sqrt( (Y(jp1) - Y(j)).^2 + (X(jp1) - X(j)).^2);
        phi = atan2((Y(jp1) - Y(j))    , (X(jp1) - X(j))    );

        A = - (x - X(j)) .* cos(phi)  - (y - Y(j)) .* sin(phi);
        B =   (x - X(j)) .^ 2         + (y - Y(j)) .^ 2;
        D =   (x - X(j));
        E =   (x - X(j)) .* sin(phi)  - (y - Y(j)) .* cos(phi);
        C = sin(-pi/2 - phi);

        u = u + lambda(j)/(2*pi) * (    C/2     .* log((S^2 + 2*A*S + B) ./ 
B) ...
                                   + (D-A*C)./E .* (atan((S+A)./E) - 
atan(A./E)));
    end % for j
end % function Uvel

%--------------------------------------------------------------------

function v = Vvel(x, y, Vinf, X, Y, lambda)
    % Vvel - y-component of velocity at (x,y)

    % uniform flow
    v = Vinf * ones(size(x));

    % loop through source panels
    n = length(lambda);
    for j = 1 : n
        if (j < n)
            jp1 = j + 1;
        else
            jp1 = 1;
        end % if

        S   = sqrt( (Y(jp1) - Y(j)).^2 + (X(jp1) - X(j)).^2);
        phi = atan2((Y(jp1) - Y(j))    , (X(jp1) - X(j))   );

        A = - (x - X(j)) .* cos(phi)  - (y - Y(j)) .* sin(phi);
        B =   (x - X(j)) .^ 2         + (y - Y(j)) .^ 2;
        D =                           + (y - Y(j));
        E =   (x - X(j)) .* sin(phi)  - (y - Y(j)) .* cos(phi);
        C = sin(- phi);

        v = v + lambda(j)/(2*pi) * (    C/2     .* log((S^2 + 2*A*S + B) ./ 
B) ...
                                   + (D-A*C)./E .* (atan((S+A)./E) - 
atan(A./E)));
    end % for j
end % function Vvel

%-------------------------------------------------------------------

function uv = strmLine(t, xy, Uinf, Vinf, X, Y, lambda)
    % strmLine - vector velocity function (used in ode45)

    % extract the x and y coordinates
    x = xy(1);
    y = xy(2);

    % get the velocities and return them as a vector
    uv = [Uvel(x, y, Uinf, X, Y, lambda); ...
         Vvel(x, y, Vinf, X, Y, lambda)];
end % function strmLine

%-------------------------------------------------------------------
```

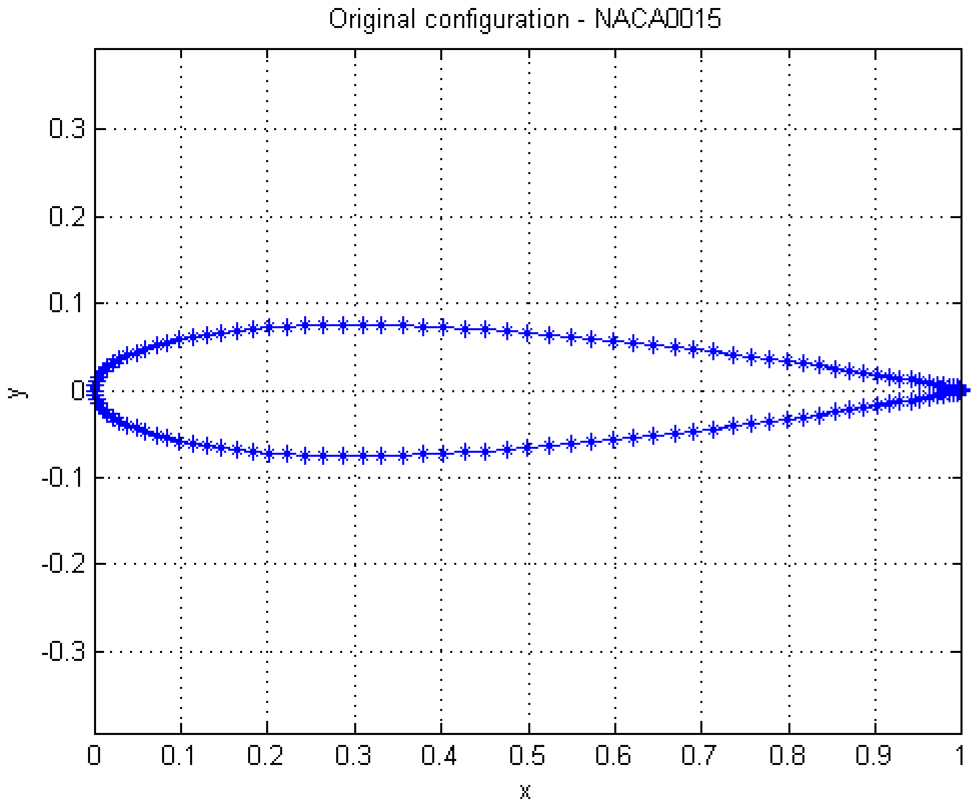
Functions

```
% Joel Lubinitsky
% AEE 342 - Project 2a: Analysis of Non-symmetric Airfoil Flows
% Integrate Cp for Cn
% 02/20/15

function area = Project2aCn(n_pan, xCpUpper, xCpLower, CpUpper, CpLower)

    areaUpper = 0;
    areaLower = 0;

    for n = [1 : (n_pan / 2) - 2]

        dxUpper   = xCpUpper(n + 1) - xCpUpper(n);
        rectUpper = CpUpper(n) * dxUpper;
        triUpper  = 0.5 * (CpUpper(n + 1) - CpUpper(n)) * dxUpper;
        areaUpper = areaUpper + rectUpper + triUpper;

        dxLower   = xCpLower(n + 1) - xCpLower(n);
        rectLower = CpLower(n) * dxLower;
        triLower  = 0.5 * (CpLower(n + 1) - CpLower(n)) * dxLower;
        areaLower = areaLower + rectLower + triLower;

    end

    area = areaLower - areaUpper;
```

```
% Joel Lubinitsky
% AEE 342 - Project 2a: Analysis of Non-symmetric Airfoil Flows
% Integrate Cp for Ca
% 02/20/15

function area = Project2aCa(n_pan, xCpUpper, xCpLower, CpUpper, CpLower,
yUpper, yLower)

    areaUpper = 0;
    areaLower = 0;

    for n = [1 : (n_pan / 2) - 2]

        dxUpper   = xCpUpper(n + 1) - xCpUpper(n);
        dydxUpper = (yUpper(n + 1) - yUpper(n)) / dxUpper;
        rectUpper = dydxUpper * CpUpper(n) * dxUpper;
        triUpper  = 0.5 * dydxUpper * (CpUpper(n + 1) - CpUpper(n)) *
dxUpper;
        areaUpper = areaUpper + rectUpper + triUpper;

        dxLower   = xCpLower(n + 1) - xCpLower(n);
        dydxLower = (yLower(n + 1) - yLower(n)) / dxLower;
        rectLower = dydxLower * CpLower(n) * dxLower;
        triLower  = 0.5 * dydxLower * (CpLower(n + 1) - CpLower(n)) *
dxLower;
        areaLower = areaLower + rectLower + triLower;

    end

    area = areaUpper - areaLower;
```

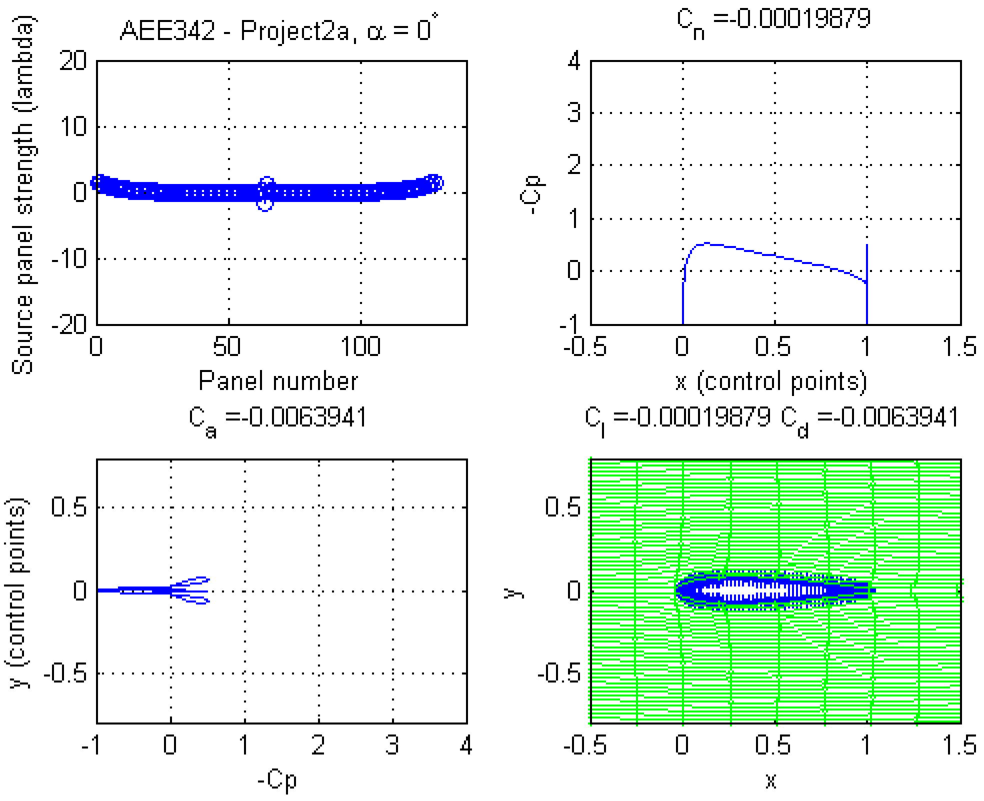
Joel Lubinitsky AEE 342
02/20/15 Project 2a

Plots



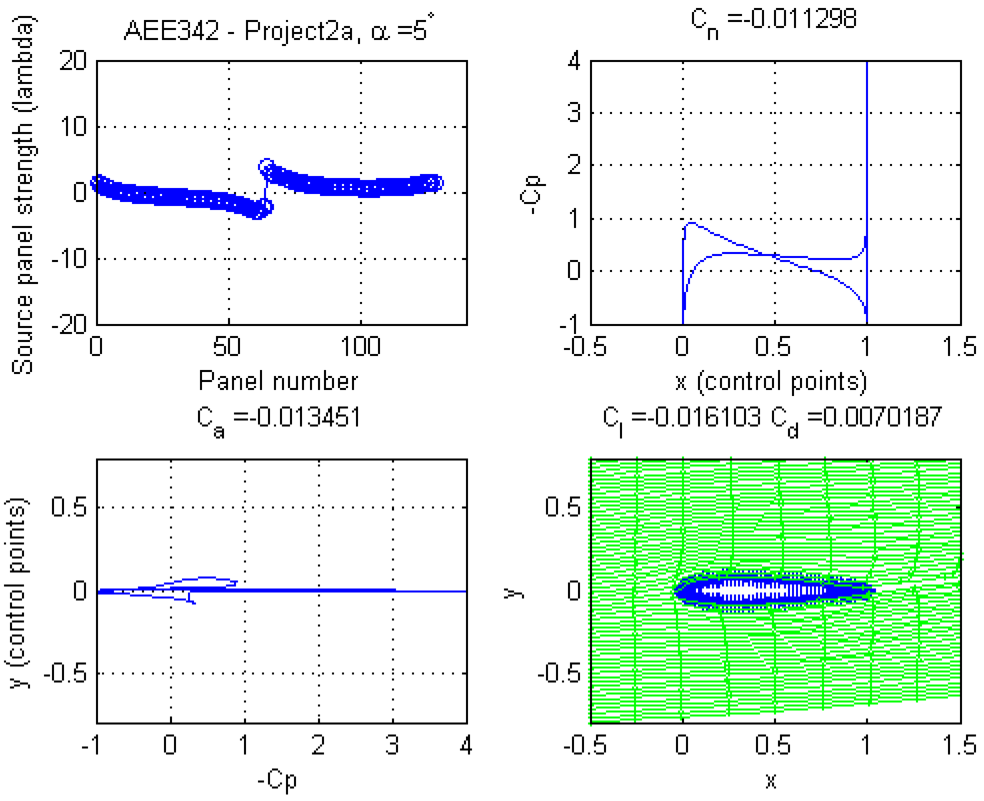
Joel Lubinitsky
02/20/15

AEE 342
Project 2a



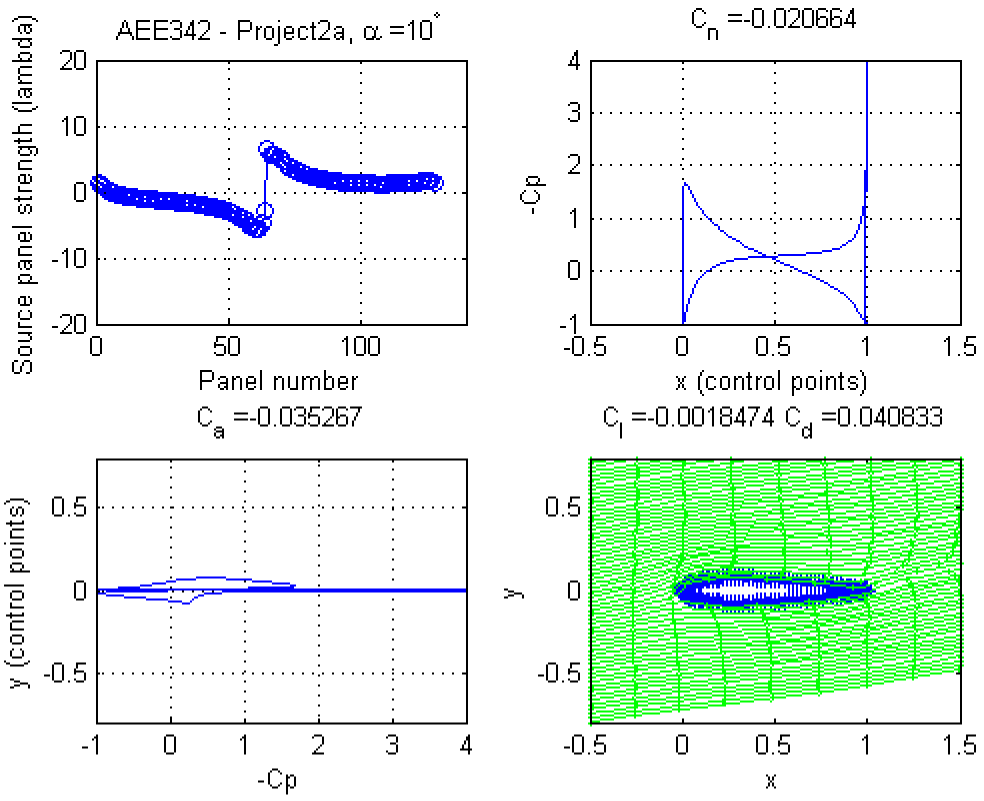
Joel Lubinitsky
02/20/15

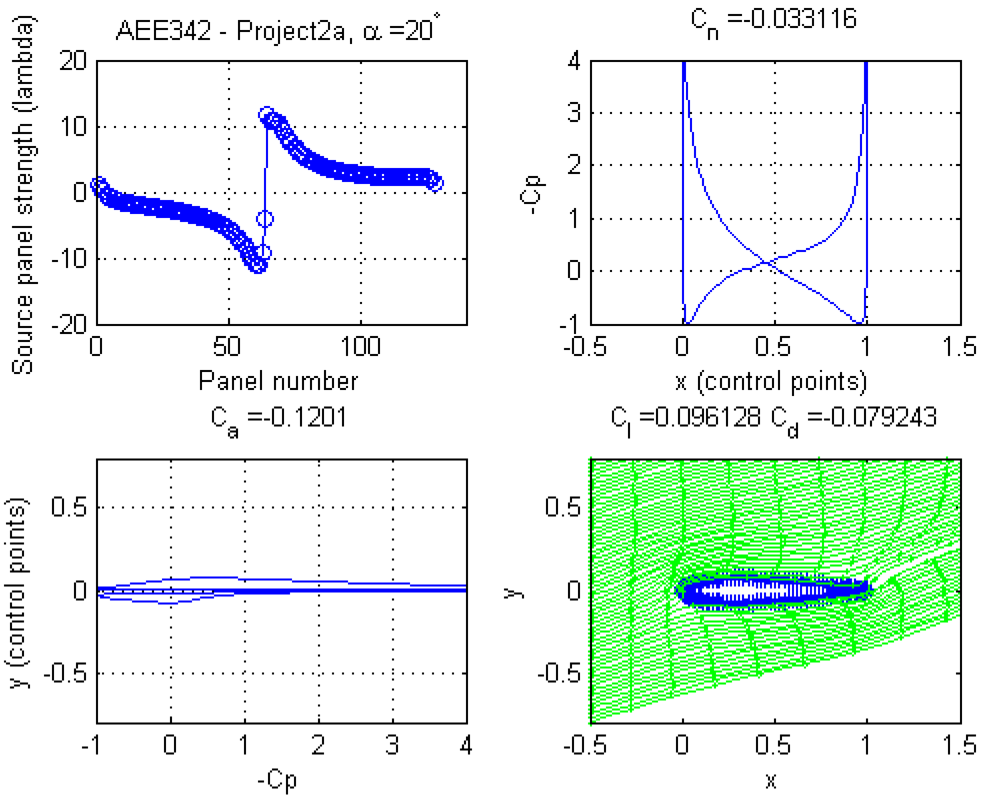
AEE 342
Project 2a



Joel Lubinitsky
02/20/15

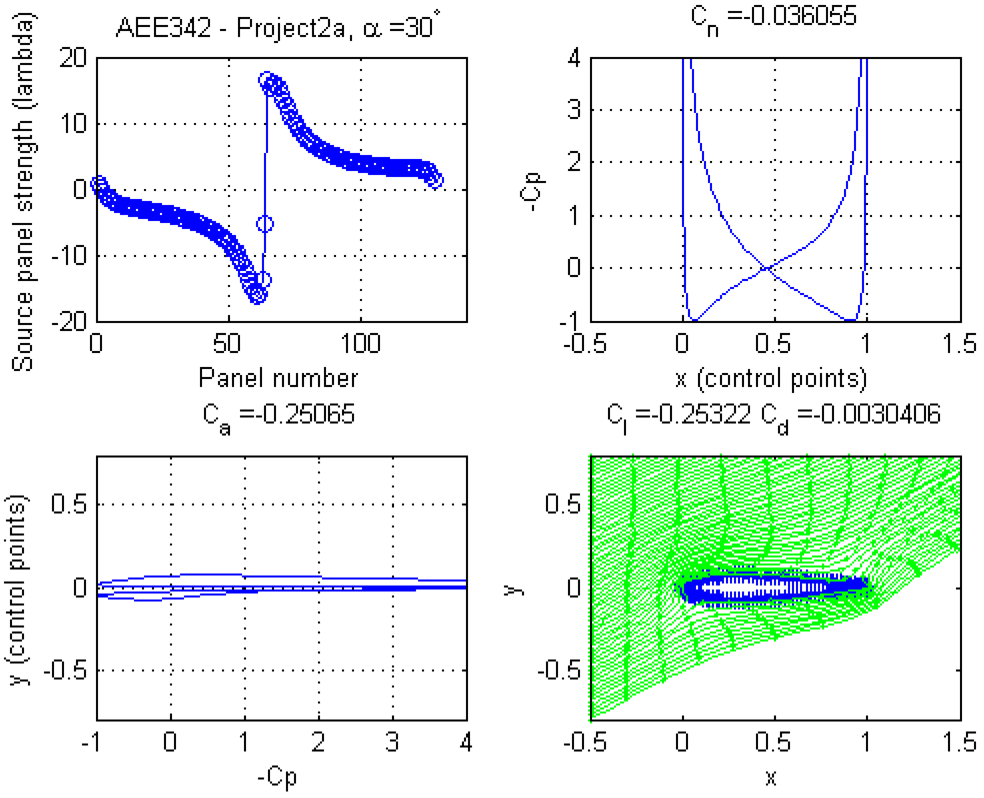
AEE 342
Project 2a

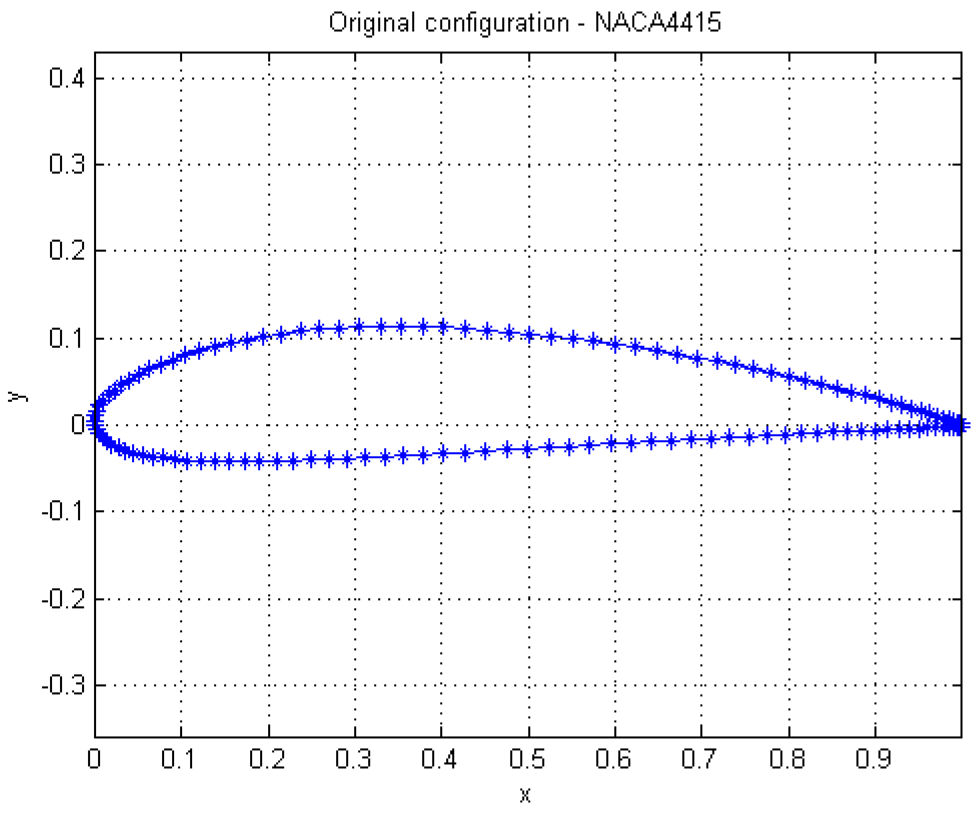



AEE342 - Project2a, α =20°
Source panel strength (lambda)
Panel number
C_n =-0.033116
-Cp
x (control points)
C_a =-0.1201
y (control points)
-Cp
C_l =0.096128 C_d =-0.079243
y
x

Joel Lubinitsky
02/20/15

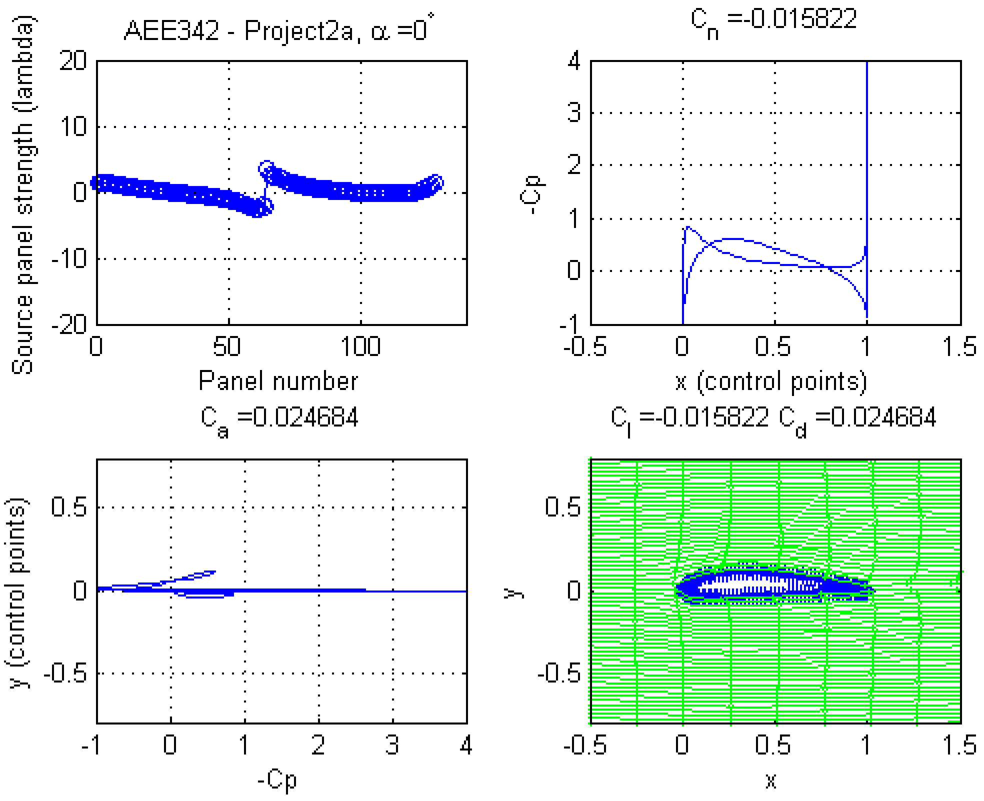
AEE 342
Project 2a

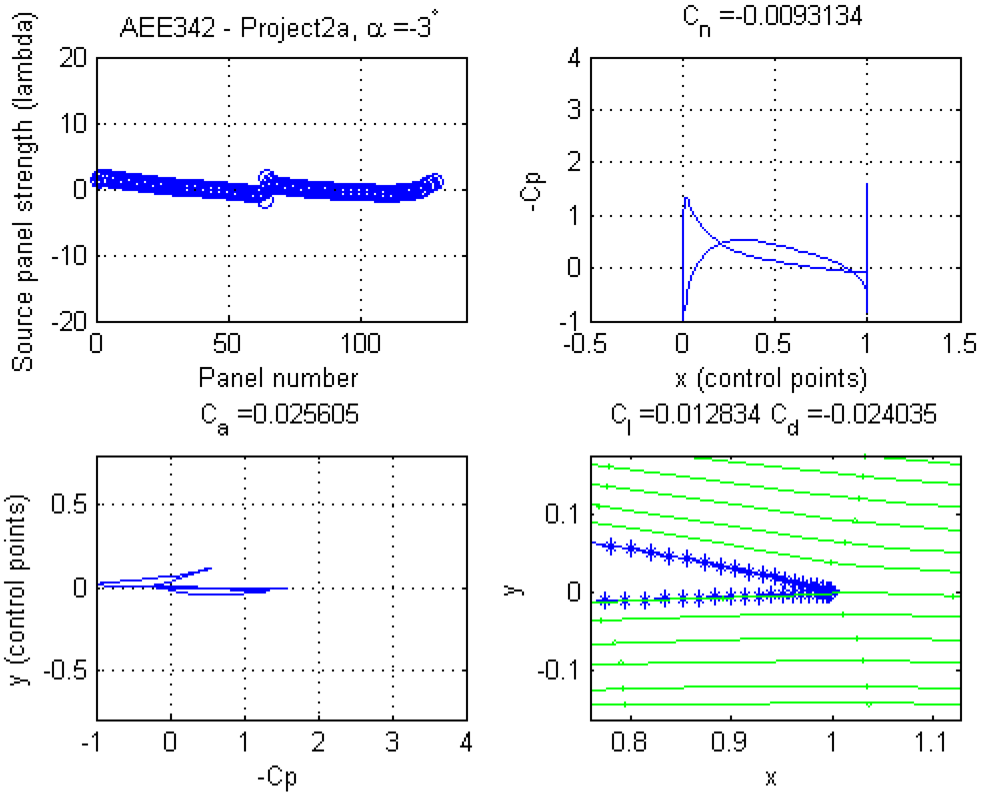



Original configuration - NACA4415
0.4
0.3
0.2
0.1
0
-0.1
-0.2
-0.3
y
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
x

Joel Lubinitsky
02/20/15

AEE 342
Project 2a




AEE342 - Project2a, α =-3°
Source panel strength (lambda)
Panel number
C_n =-0.0093134
-Cp
x (control points)
C_a =0.025605
y (control points)
-Cp
C_l =0.012834 C_d =-0.024035
y
x

References

Matlab Documentation. http://www.mathworks.com/help/matlab/.