```
%==========================================================================
%
% VORTEX PANEL METHOD CODE FOR MAE 424: AERODYNAMICS
%
% 2nd-order panel method, i.e. uses panels with linearly-varying strength.
% First part of code creates (x,y) coordinates for NACA 4-digit airfoils.
% Second part calculates the lift coefficient, etc.
% An overall 'for' loop over k produces variations with angle of attack.
%
% Originally created for MATLAB under the supervision of Dr. Cyrus Madnia
% Revised in April 2015 and 2017 by Dr. Matthew Ringuette
%
% Based on the panel method code published in:
% Kuethe & Chow "Foundations of Aerodynamics: Bases of
% Aerodynamic Design" 1998, John Wiley & Sons
%
% That code based on: Stevens, W. A., Goradia, S. H., & Braden, J. A.,
% "Mathematical Model for Two-Dimensional Multi-Component Airfoils in
% Viscous Flow," NASA CR-1843, 1971.
%
% The airfoil geometry information is based on both Kuethe & Chow and:
% Ladson, C. L., Brooks, Jr., C. W., & Hill, A. S., "Computer Program to
% Obtain Ordinates for NACA Airfoils," NASA TM 4741, 1996.
%
%==========================================================================

%close all
clear all
clc

disp('Running MAE 424 vortex panel method code.');
disp(' ');

%==========================================================================
% Part I: user input (airfoil geometry, # panels, min/max angles of attack)
%==========================================================================

Npfig = figure(3);
NACAfig = figure(5);
alphafig = figure(8);

pmxx = input('Enter desired NACA 4-digit designation, e.g. 0012: ');
disp(' ');
Np = input('Enter number of vortex panels (even integer, ideally 40-100): ');
disp(' ');
alpha_min = input('Enter the minimum angle of attack to test (in degrees): ');
disp(' ');
alpha_max = input('Enter the maximum angle of attack to test (in degrees): ');
disp(' ');
kmax = input('Enter the max number of angles to test over this range (integer): ');
disp(' ');

%==========================================================================
% Part II: creation of NACA 4-digit airfoil panel geometry
%
% Calculates the coordinates of the airfoil panel boundary points
% (xfoil,yfoil), and coordinates of the mean camber line points (xc,yc)
%==========================================================================

disp('Creating (x,y) coordinates of airfoil boundary points, camber line...');
disp(' ');

% Initialize vectors used
%~~~~~~~~~~~~~~~~~~~~~~~~

xc(1:Np+1) = 0; % Original boundary-point x/c values
yt(1:Np+1) = 0; % Thickness y/c values
yc(1:Np+1) = 0; % Camber y/c values
xfoil(1:Np+1) = 0; % Final boundary-point x/c values
yfoil(1:Np+1) = 0; % Final boundary-point y/c values

% Isolate key parameters of the user-specified airfoil geometry
% ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

pm = floor(pmxx/100);
xx = pmxx-pm*100; % Thickness in percent chord
p = floor(pm/10); % Camber in percent chord
m = pm-p*10; % Location of max camber in chord lengths*10
```

```
% Code portion below ending with "name" is used for plot title below. The
% "if" statements handle all possible 4-digit cases, since "int2str" (and
% "num2str" if used) has problems with converting leading zeros.
if p == 0
    pstring = '0';
else
    pstring = int2str(p);
end
if m == 0
    mstring = '0';
else
    mstring = int2str(m);
end
if xx == 0
    xxstring = '00';
elseif xx < 10
    xxstring = ([‘0’,int2str(xx)]);
else
    xxstring = int2str(xx);
end
name = ([pstring,mstring,xxstring]);

xx = xx/100; % Rescale thickness to be in chord lengths
p = p/100; % Rescale max camber to be in chord lengths
m = m/10; % Rescale max camber location to be in chord lengths

% Compute the aifoil thickness distribution
% ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

% ***NOTE: ALL DISTANCES IN THIS CODE ARE DIMENSIONLESS, NONDIMENSIONALIZED
% BY THE CHORD LENGTH, I.E. IN TERMS OF CHORD LENGTHS

% Next, find the x/c boundary-point locations for a zero-camber airfoil
% using the "circle" method, which ultimately clusters points near the
% leading (x/c=0) and trailing (x/c=1) edges. The boundary-point numbering
% should start at the TE then go clockwise around the airfoil and back to
% the TE.  So, x/c(1) = 1, xc(Np/2+1) = 0, xc(Np+1) = 1. For each circle
% boundary point at an angle beta, use trig to find the intersection with
% the x-axis, which we call xc.
for i = (Np+1):-1:1
    beta = (2*pi)/Np*(i-1); % In radians; (2*pi)/Np produces equal interval
                            % angles around the circle
    xc(i) = 0.5*(1+cos(beta)); % The center of the circle is at x/c = 0.5,
                               % so need to add 0.5 to x/c values
end

% Calculate the y/c values of the thickness distribution versus x/c
% First, find the polynomial coefficients for the thickness distribution
% from the NASA report
acoef = xx/0.20*[0.2969 -0.1260 -0.3516 0.2843 -0.1015];

for i = 1:(Np+1)
    if i <= (Np/2) % Points from the TE to 1 point from the LE (lower
                   % boundary)
        flip = -1;
    else % Points from the LE to the TE (upper boundary)
        flip = 1;
    end
    yt(i) = flip*(acoef(1)*sqrt(xc(i))+acoef(2)*xc(i)+...
        acoef(3)*(xc(i))^2+acoef(4)*(xc(i))^3+acoef(5)*(xc(i))^4);
end

% Wrap thickness around camber line, get final airfoil boundary points
% ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

% First compute some factors needed for the camber line
if p~=0 % If nonzero camber
  fact1 = p/m^2; fact2 = 2*m; fact3 = p/(1-m)^2;
else % If zero camber
  fact1 = 0; fact2 = 0; fact3 = 0; % fact2 can be any value, just needs one
end

% Next, in one "for" loop, calculate the y/c values of the camber line
% versus x/c, and also calculate the final airfoil boundary points by
% "wrapping" the thickness envelope around the camber line, one point at a
% time, i.e. for each x/c--this requires a geometric transformation, given
% in the NASA report.  Note: the camber line coordinates define two of the
```

```
% same camber line: from TE to LE, and LE to TE, for convenience of use in
% determining xfoil and yfoil in the "for" loop.
for i=1:(Np+1)

   if xc(i)<m % Forward of the maximum camber point
     yc(i) = fact1*(fact2*xc(i)-(xc(i))^2); % y/c location of camber line
     tan_delta  = fact1*(fact2-2*xc(i)); % Local slope of camber line

   else % At or aft of the maximum camber point
     yc(i) = fact3*(1-fact2+xc(i)*fact2-(xc(i))^2); % y/c location of
                                                    % camber line
     tan_delta  = fact3*(fact2-2*xc(i)); % Local slope of camber line

   end

   cos_delta = 1/sqrt(1+tan_delta^2); % Standard trig relation
   sin_delta = tan_delta*cos_delta;

   % Lastly, compute the new x/c and y/c coordinates for the arfoil panels
   xfoil(i) = xc(i)-yt(i)*sin_delta;
   yfoil(i) = yc(i)+yt(i)*cos_delta;

end

% Plot airfoil panel geometry (via the boundary points) to check the result
% ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~

figure(1)
% Lower airfoil boundary
plot(xfoil(1:(Np/2+1)),yfoil(1:(Np/2+1)),'r','LineWidth',1);
hold on
% Upper airfoil boundary
plot(xfoil((Np/2+1):(Np+1)),yfoil((Np/2+1):(Np+1)),'g','LineWidth',1);
% Camber line, just the single curve from LE to TE
plot(xc(Np/2+1:Np+1),yc(Np/2+1:Np+1),'b','LineWidth',1);

title(['NACA ',name,' Airfoil with ',int2str(Np),' Panels'],'FontSize',15);
xlabel('x/c','FontSize',14);
ylabel('y/c','FontSize',14);
xlim([0 1]);
daspect([1 1 1]); % Make sure x- and y-axes have same scale factor
grid on;
box on;

%=========================================================================
% Part III: calculations using vortex panel method
%
%    Calculates:  gamma - array of circulation densities;
%                 xcen  - ordinates of panel centers;
%                 cp    - pressure coefficient;
%                 clift - lift coefficient;
%                 cdrag - drag coefficient (aerodynamic);
%                 ista - starting panel i index;
%                 iend - ending panel i index;
%                 xcp - center of pressure;
%                 cmle - moment coeff. about leading edge;
%                 cmqc - moment coeff. about 1/4 chord.
%
%    All notations follow Kuethe & Chow (1998)
%=========================================================================

disp('Running vortex panel method...');
disp(' ');


Np1 = Np+1;

alpha_min = alpha_min*pi/180;
alpha_max = alpha_max*pi/180;
alpha = linspace(alpha_min,alpha_max,kmax)'; % Angle of attack vector
alpha_deg = alpha*180/pi;

% Initialize column vectors of calculated quantities, fill with 0's
clift = zeros(kmax,1);
cdrag = zeros(kmax,1);
xcp = zeros(kmax,1);
cmle = zeros(kmax,1);
cmqc = zeros(kmax,1);
```

```
for k = 1:kmax % Overall 'for' loop for varying angle of attack

%
%  Compute geometrical characteristics
%
for i=1:Np,
    ip1 = i+1;
        x(i) = 0.5*(xfoil(i)+xfoil(ip1));
        y(i) = 0.5*(yfoil(i)+yfoil(ip1));
        s(i) = sqrt((xfoil(i)-xfoil(ip1))^2+...
                    (yfoil(i)-yfoil(ip1))^2);
        theta(i) = atan2((yfoil(ip1)-yfoil(i)),...
                         (xfoil(ip1)-xfoil(i)));
        sine(i) = sin(theta(i)); cosine(i) = cos(theta(i));
        rhs(i) = sin(theta(i)-alpha(k));
end
%
%  Assemble the system matrix
%
for i=1:Np, for j=1:Np
   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 =-(x(i)-xfoil(j))*cosine(j)-(y(i)-yfoil(j))*sine(j);
     B = (x(i)-xfoil(j))^2+(y(i)-yfoil(j))^2;
     C = sin(theta(i)-theta(j));
     D = cos(theta(i)-theta(j));
     E = (x(i)-xfoil(j))*sine(j)-(y(i)-yfoil(j))*cosine(j);
     F = log(1+s(j)*(s(j)+2*A)/B);
     G = atan2(E*s(j),B+A*s(j));
     P = (x(i)-xfoil(j))*sin(theta(i)-2*theta(j))+...
         (y(i)-yfoil(j))*cos(theta(i)-2*theta(j));
     Q = (x(i)-xfoil(j))*cos(theta(i)-2*theta(j))-...
         (y(i)-yfoil(j))*sin(theta(i)-2*theta(j));
     cn2(i,j) = D+0.5*Q*F/s(j)-(A*C+D*E)*G/s(j);
     cn1(i,j) = 0.5*D*F+C*G-cn2(i,j);
     ct2(i,j) = C+0.5*P*F/s(j)+(A*D-C*E)*G/s(j);
     ct1(i,j) = 0.5*C*F-D*G-ct2(i,j);
   end
end, end
%
%
%
for i=1:Np
   An(i,1)   = cn1(i,1); An(i,Np1) = cn2(i,Np);
   At(i,1)   = ct1(i,1); At(i,Np1) = ct2(i,Np);
   for j=2:Np
      An(i,j)=cn1(i,j)+cn2(i,j-1); At(i,j)=ct1(i,j)+ct2(i,j-1);
   end
end
An(Np1,1) = 1; An(Np1,Np1) = 1;
for j=2:Np, An(Np1,j) = 0; end
rhs(Np1) = 0;
%
%  Solve for vortex panel strengths
%
gamma = An\rhs';
%
xcen = x;
i=1; while xcen(i+1)>0.99, i=i+1; end
ista = i;
i=Np; while xcen(i-1)>0.99, i=i-1; end
iend = i;
%
%  Compute Cp, Clift, and Cdrag
%
cx = 0; cy = 0; cmle(k) = 0; cmqc(k) = 0;
for i=ista:iend
   v(i) = cos(theta(i)-alpha(k));
   for j=1:Np1
      v(i) = v(i)+At(i,j)*gamma(j);
   end
   cp(i) = 1-v(i)^2;
   cx = cx+cp(i)*s(i)*sine(i);
   cy = cy-cp(i)*s(i)*cosine(i);
   cmle(k) = cmle(k)+cp(i)*s(i)*cosine(i)*xcen(i);
   cmqc(k) = cmqc(k)+cp(i)*s(i)*cosine(i)*(xcen(i)-0.25);
```

```
end
clift(k) = cy*cos(alpha(k))-cx*sin(alpha(k));
cdrag(k) = cy*sin(alpha(k))+cx*cos(alpha(k));
xcp(k) = -cmle(k)/clift(k);

end % for k = 1:kmax

% NACA 2412 experimental data at Re = 3.1 million, from Abbott, I. H. and
% Von Doenhoff, A. E., "Theory of Wing Sections," Dover, 1959
clift_exp = [-0.185 0.04 0.225 0.445 0.645 0.86 1.065 1.255 1.43 1.585 1.59 1.545 1.505 1.215]';
alpha_exp = [-4.2 -2.0 0.1 2.1 4.1 6.0 8.0 10.0 12.0 14.1 15.1 16.1 16.3 18.3]';

% ***Basic*** plot of lift coefficient vs. angle of attack
% Question 1
figure(2)
plot(alpha_deg,clift,'r','Linewidth',1);
hold on;
plot(alpha_deg,cdrag,'g','Linewidth',1);
plot(alpha_deg,cmqc,'b','Linewidth',1);
plot(alpha_exp,clift_exp,'k','Linewidth',1);
title('Coefficients vs Angle of Attack');
xlabel('Angle of Attack (degrees)');
ylabel('Lift, Drag, Moment about 1/4 chord and Experimental Lift Coefficients');
legend('Lift Coefficient','Drag Coefficient','Moment Coefficient about 1/4 chord','Experimental Lift
Coefficient');
% Question 1 part f
figure(Npfig)
plot(alpha_deg,clift,'b','Linewidth',1);
title('Coefficient of Lift vs Angle of Attack for Varying Number of Panels');
xlabel('Angle of Attack (degrees)');
ylabel('Lift Coefficient');
legend('6 panels','26 panels','100 panels');
%Question 2 Part A
figure(NACAfig);
title('Coefficients of Lift and Moment about 1/4 Chord vs Angle of Attack');
xlabel('Angle of Attack (degrees)');
ylabel('Coefficients of Lift and Moment about 1/4 Chord');
legend('Coefficient of Lift for NACA 2812','Coefficient of Moment for NACA 2812','Coefficient of Lift
for NACA 2612','Coefficient of Moment for NACA 2612','Coefficient of Lift for NACA 2412','Coefficient
of Moment for NACA 2412')
plot(alpha_deg,clift,'r','Linewidth',1);
hold on;
plot(alpha_deg,cmqc,'b','Linewidth',1);
%Question 2 Part B
figure(NACAfig);
title('Coefficients of Lift and Moment about 1/4 Chord vs Angle of Attack');
xlabel('Angle of Attack (degrees)');
ylabel('Coefficients of Lift and Moment about 1/4 Chord');
legend('Coefficient of Lift for NACA 2424','Coefficient of Moment for NACA 2424','Coefficient of Lift
for NACA 2418','Coefficient of Moment for NACA 2418','Coefficient of Lift for NACA 2412','Coefficient
of Moment for NACA 2412')
plot(alpha_deg,clift,'r','Linewidth',1);
hold on;
plot(alpha_deg,cmqc,'b','Linewidth',1);
% Question 3 part a
figure(6);
plot(alpha_deg,clift,'r','Linewidth',1);
xlabel('Angle of Attack (degrees)');
ylabel('Coefficient of Lift');
title('Coefficient of Lift vs Angle of Attack');
% Question 3 part b
figure(7);
plot(alpha_deg,xcp,'m','Linewidth',1);
xlabel('Angle of Attack (degrees)');
ylabel('Center of Pressure');
title('Center of Pressure vs Angle of Attack');
% Question 3 part c
figure(alphafig);
plot(xcen(ista:iend),cp(ista:iend),'b','Linewidth',1);
xlabel('x-locations of the panel centers');
ylabel('Pressure Coefficient');
title('Pressure Coefficient vs x-locations of Panel Centers');
legend('Alpha = -2 degrees','Alpha = 5 degrees');
```